(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号
特開2002－159009
（P2002－159009A）
(43)公開日　平成14年５月31日(2002.5.31)

(51)Int.Cl.[7]		識別記号	ＦＩ		テーマコード（参考）
Ｈ０４Ｎ	7/30		Ｈ０３Ｍ	7/30	Ａ　５Ｃ０５９
Ｈ０３Ｍ	7/30			7/40	５Ｊ０６４
	7/40		Ｈ０４Ｎ	7/133	Ｚ

審査請求　未請求　請求項の数38　ＯＬ　（全 40 頁）

(21)出願番号　特願2001－261489(P2001－261489)

(22)出願日　平成13年８月30日(2001.8.30)

(31)優先権主張番号　ＰＱ９８２４
(32)優先日　平成12年９月１日(2000.9.1)
(33)優先権主張国　オーストラリア（ＡＵ）

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号
(72)発明者　ドミニック　イップ
オーストラリア国　2113　ニュー　サウス　ウェールズ州，　ノース　ライド，　トーマス　ホルト　ドライブ　１　キヤノン　インフォメーション　システムズ　リサーチ　オーストラリア　プロプライエタリー　リミテツド　内
(74)代理人　100076428
弁理士　大塚　康徳　（外３名）

最終頁に続く

(54)【発明の名称】　エントロピ符号化及び復号化方法とその装置

(57)【要約】　（修正有）
【課題】　デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する方法が提供される。
【解決手段】　シンボルをエントロピ符号化するための、シグニフィカンス・プロパゲーション・パス３１４と、マグニチュード・リファインメント・パス３１６と、クリーンアップ・パス３１８とを有する。現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中エントロピ符号化の対象となるシンボルを持つコード・ブロック内の位置の第１の係数リストを生成し、コード・ブロック内の前記係数である、現在のビットプレーンのマグニチュード・リファインメント・パス中エントロピ符号化の対象となるシンボルを持つ係数の位置の第２のリストも生成し、コード・ブロック内の前記係数である、現在のビットプレーンのクリーンアップ・パス中にエントロピ符号化の対象となるシンボルを持つ係数の位置の第３のリストをさらに生成する。

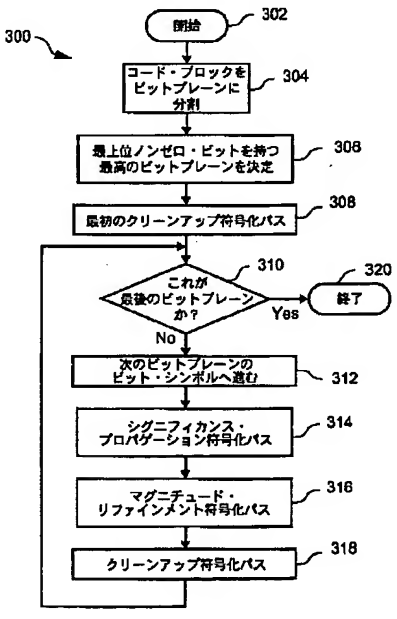


Fig. 3

1

【特許請求の範囲】

【請求項１】　デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する方法であって、第１のビットプレーンから所定の最下位ビットプレーンまでの各ビットプレーンについて、

現在のビットプレーン内にゼロの有意状態を持つ係数であって、現在のビットプレーン内に１の有意状態を持つ近傍係数を持つ係数に属するシンボルである、現在のビットプレーン内の全てのシンボルのエントロピ符号化を行うシグニフィカンス・プロパゲーション(significance propagation)ステップと、

前回のビットプレーン内に１の有意状態を持つ係数に属するシンボルである、現在のビットプレーン内の全てのシンボルのエントロピ符号化を行うマグニチュード・リファインメント・ステップと、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ又は前記マグニチュード・リファインメント・ステップ中、それ以前にエントロピ符号化されなかったシンボルである、現在のビットプレーン内の全てのシンボルのエントロピ符号化を行うクリーンアップ・ステップとを実行し、

更に、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第１のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップに先行して生成するステップと、

現在のビットプレーンの前記マグニチュード・リファインメント・ステップ中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第２のリストを、現在のビットプレーンの前記マグニチュード・リファインメント・ステップに先行して生成するステップと、

現在のビットプレーンの前記クリーンアップ・ステップ中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第３のリストを、現在のビットプレーンの前記クリーンアップ・ステップに先行して生成するステップと、を有することを特徴とする方法。

【請求項２】　請求項１に記載の方法において、前記第１及び第２のリストを生成する前記ステップが、前回のビットプレーンの前記クリーンアップ・ステップ中行われることを特徴とする方法。

【請求項３】　請求項１に記載の方法において、前記第３のリストを生成する前記ステップが、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中に行われることを特徴とする方法。

【請求項４】　請求項１に記載の方法において、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップは、

前回のビットプレーンの前記クリーンアップ・ステップによって生成される前記第１のリスト内の前記位置に対応するシンボルである、現在のビットプレーン内のシンボルのエントロピ符号化を行うサブステップと、

ゼロの非有意状態を持つ、現在のビットプレーン内の任意のシンボルであって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中に１の有意状態に変化する近傍として、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中、予めエントロピ符号化されたシンボルを持つ係数を有する任意のシンボルである、所定の走査順で前記近傍に後続する任意のシンボルのエントロピ符号化を行うサブステップと、

ゼロの非有意状態を持つ、コードブックの係数の位置と、１の有意状態を持つ現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップでエントロピ符号化されるシンボルを有する前記係数に隣接する係数の位置の第４のリストを生成するサブステップと、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中、１の有意状態に変化する前記係数の前記第１のリストの前記位置にタグを付けるサブステップと、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中、１の有意状態に変化する前記係数の前記第４のリストの前記位置にタグを付けるサブステップと、

前記コード・ブロック内の係数の位置の前記第３のリストを生成するサブステップと、を有することを特徴とする方法。

【請求項５】　請求項４に記載の方法において、

前記クリーンアップ・ステップは、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップによって生成されたシンボルであって、前記第３のリスト内の前記位置に対応するシンボルである、現在のビットプレーン内のシンボルのエントロピ符号化を行うサブステップと、

現在のビットプレーンの前記クリーンアップ・ステップ中、エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンの前記クリーンアップ中、１の有意状態に変化する係数である、前記コード・ブロック内の前記係数の位置の第５のリストを生成するサブステップと、

現在のビットプレーンの前記クリーンアップ・ステップ中、エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンの前記クリーンアップで、ゼロの有意状態を持つ係数である、前記コード・ブロック内の前記係数の位置の第６のリストを生成するサブステップと、を有することを特徴とする方法。

【請求項６】　請求項５に記載の方法において、

前記クリーンアップ・ステップは、
前記第１のリストを前記タグを付けられた位置の第７のリストと、タグを付けられなかった位置の第８のリストとにソートするサブステップと、
前記第４のリストを、前記タグを付けられた位置の第９のリストと、タグを付けられなかった位置の第１０のリストとにソートするサブステップと、を有することを特徴とする方法。
【請求項７】　請求項６に記載の方法において、
現在のビットプレーンが使用される前記クリーンアップ・ステップは、
前記第２、第５、第７、第９のリストの前記位置を比較し併合して、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成するサブステップと、
前記第６、第８、第１０のリストの前記位置を比較し併合して、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストを生成するサブステップと、
前記生成された第１のリストから、１の有意状態を持つ係数である、前記コード・ブロック内の係数の任意の位置を除去するサブステップと、を有することを特徴とする方法。
【請求項８】　請求項１に記載の方法において、前記第１のビットプレーンは、有意係数を持つ前記コード・ブロックの最上位ビットプレーンより下の次のビットプレーンであることを特徴とする方法。
【請求項９】　請求項８に記載の方法において、
前記コード・ブロックの前記最上位ビットプレーン内の全てのシンボルをエントロピ符号化するの最初のクリーンアップ・ステップを有することを特徴とする方法。
【請求項１０】　請求項９に記載の方法において、
前記最初のクリーンアップ・ステップは、
前記第１のビットプレーンに対する前記シグニフィカンス・プロパゲーション・ステップで、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の係数の位置の前記第１のリストを生成するサブステップと、
前記第１のビットプレーンに対する前記マグニチュード・リファインメント・ステップで、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の係数の位置の前記第２のリストを生成するサブステップと、を有することを特徴とする方法。
【請求項１１】　請求項９に記載の方法において、
前記最初のクリーンアップ・ステップは、
前記最初のクリーンアップ・ステップ符号化パス中、符号化された前記各シンボルについて、前記エントロピ符号化されたシンボルが属する前記係数の前記有意状態を決定するサブステップであって、前記有意状態が１の場合、
第１のフラグをセットするサブステップと、
前記所定の１の有意状態を有する前記係数の前記近傍係数を決定し、前記各近傍係数について第２のフラグを１にセットするサブステップとを実行するサブステップと、
一方、前記所定の有意状態がゼロの場合、
前記第１のフラグをゼロにセットするサブステップと、
前記第１のビットプレーンが使用される前記マグニチュード・リファインメント・ステップ中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の係数の位置の前記第２のリストを生成するサブステップであって、前記位置に関連する前記第１のフラグが１の場合、前記係数の前記位置が前記第２のリストに追加されるように構成されるサブステップと、
前記第１のビットプレーンが使用される前記シグニフィカンス・プロパゲーション・ステップ中、エントロピ符号化の対象となる前記シンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の前記第１のリストを生成するサブステップであって、前記位置に関連する前記第２のフラグが１であり、かつ、前記位置に関連する前記第１のフラグがゼロである場合、前記係数の前記位置を前記第１のリストに追加するように構成されるサブステップと、を有することを特徴とする方法。
【請求項１２】　請求項１に記載の方法において、前記エントロピ符号化法が、前記シンボルのエントロピ符号化または復号化を行うことが可能であることを特徴とする方法。
【請求項１３】　請求項１に記載の方法において、前記エントロピ符号化法が前記シンボルのエントロピ符号化を行う方法であって、現在のビットプレーン用の前記第１と、第２と、第３のリストを生成する前記ステップが、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップに先行して生成されることを特徴とする方法。
【請求項１４】　請求項１３に記載の方法において、現在のビットプレーンの前記第２のリストを生成する前記方法は、
現在のビットプレーン番号Ｎ以上の有意状態を持つ前記第２のリストに前記コード・ブロック内の係数の前記位置を追加するサブステップを有することを特徴とする方法。
【請求項１５】　請求項１３または１４に記載の方法であって、現在のビットプレーンの前記第１のリストを生成する前記方法において、
現在のビットプレーン番号Ｎ未満の有意状態を持つ係数であって、所定の走査順でＮ以上の有意状態を持つ近傍係数係数に先行する係数である、前記コード・ブロック内の任意の係数の前記位置を前記第１のリストに追加するサブステップを有することを特徴とする方法。
【請求項１６】　デジタル画像の変換係数を有するコー

ド・ブロックを表すシンボルをエントロピ符号化する装置において、
第１のビットプレーンから所定の最下位ビットプレーンへ、シンボルから成るビットプレーンに前記コード・ブロックを分割するビットプレーン・スプリッタと、
シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中に、前記シンボルを符号化するエントロピ符号器と、
現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に、エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第１のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パスに先行して生成し、現在のビットプレーンのマグニチュード・リファインメント・パス中に、エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第２のリストを、現在のビットプレーンの前記マグニチュード・リファインメント・パスに先行して生成し、現在のビットプレーンのクリーンアップ・パス中に、エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第３のリストを、現在のビットプレーンの前記クリーンアップ・パスに先行して生成するリスト・メモリ・マネージャと、
前記第１の位置リストと、第２の位置リストと、第３の位置リストとを格納する記憶ユニットと、を有することを特徴とする装置。
【請求項１７】　請求項１６に記載の装置において、
前記記憶ユニットは、
前記リスト・メモリ・マネージャによってビットプレーンｎの前記第１のリストを入力して格納し、さらに、既に生成され、格納されたビットプレーン（ｎ＋１）の前記第１のリストを同時に出力するダブルバッファとなるように構成される第１の記憶ユニットと、
前記リスト・メモリ・マネージャによってビットプレーンｎの前記第２のリストを入力し、格納し、更に、既に生成され格納されたビットプレーン（ｎ＋１）の前記第２のリストを同時に出力するダブルバッファとなるように構成される第２の記憶ユニットと、
前記リスト・メモリ・マネージャによってビットプレーンｎの前記第３のリストを入力し、格納し、更に、既に生成され格納されたビットプレーン（ｎ＋１）の前記第３のリストを同時に出力するダブルバッファとなるように構成される第３の記憶ユニットと、を有することを特徴とする装置。
【請求項１８】　請求項１６に記載の装置において、前記リスト・メモリ・マネージャは、前回のビットプレーンのクリーンアップ・パス中に前記シンボルの前記エントロピ符号化を行っている間、前記第１及び第２のリストを生成することを特徴とする装置。
【請求項１９】　請求項１６に記載の装置において、前記リスト・メモリ・マネージャは、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に前記シンボルの前記エントロピ符号化を行っている間、前記第３のリストを生成することを特徴とする装置。
【請求項２０】　請求項１６に記載の装置において、
前記エントロピ符号器は、
前回のビットプレーンの前記クリーンアップ・パスによって生成された前記第１のリスト内の前記位置に対応する現在のビットプレーン内の前記シンボルのエントロピ符号化を、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に行う手段と、
ゼロの非有意状態を持つ任意のシンボルであって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する近傍として予めエントロピ符号化されたシンボルを持つ近傍係数を有する任意のシンボルである、所定の走査順で前記近傍に後続する任意のシンボルのエントロピ符号化を、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に現在のビットプレーン内で行う手段と、を有し、
前記リスト・メモリ・マネージャは、
ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップ中、エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを生成する手段と、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に、１の有意状態に変化する前記係数の前記第１のリストの位置にタグを付ける手段と、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に、１の有意状態に変化する前記係数の前記第４のリストの位置にタグを付ける手段と、
前記コード・ブロック内の係数の位置の前記第３のリストを生成する手段と、を有することを特徴とする装置。
【請求項２１】　請求項２０に記載の装置において、
前記エントロピ符号器は、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パスによって生成されたシンボルである、前記第３のリスト内の前記位置に対応する現在のビットプレーン内のシンボルのエントロピ符号化を現在のビットプレーンのクリーンアップ・パス中に行う手段を有し、
前記リスト・メモリ・マネージャは、
現在のビットプレーンの前記クリーンアップ・パス中、エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンの前記クリーンアップ中、１の有意状態に変化する係数である、前記コード・ブロック内

の前記係数の位置の第５のリストを生成する手段と、
現在のビットプレーンの前記クリーンアップ・パス中にエントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンの前記クリーンアップ中ゼロの有意状態を持つ係数である、前記コード・ブロック内の前記係数の位置の第６のリストを生成する手段と、を有することを特徴とする装置。

【請求項２２】　請求項２１に記載の装置において、
前記リスト・メモリ・マネージャは、
タグを付けられた第７の位置リストと、タグを付けられなかった第８の位置リストとに前記第１のリストをソートする手段と、
タグを付けられた第９の位置リストと、タグを付けられなかった第１０の位置リストとに前記第４のリストをソートする手段と、を有することを特徴とする装置。

【請求項２３】　請求項２２に記載の装置において、
前記リスト・メモリ・マネージャは、
前記第２、第５、第７、第９のリストの前記位置を比較し併合して、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成する手段と、
前記第６、第８、第１０のリストの前記位置を比較し併合して、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストを生成する手段と、
前記生成された第１のリストから、１の有意状態を持つ係数である、前記コード・ブロック内の係数の任意の位置を取り除く手段と、を有することを特徴とする装置。

【請求項２４】　請求項１６に記載の装置において、前記第１のビットプレーンが、有意係数を持つ前記コード・ブロックの最上位ビットプレーンより下の次のビットプレーンであることを特徴とする装置。

【請求項２５】　請求項２４に記載の装置において、
前記エントロピ符号器は、最初のクリーンアップ・ステップ・パス中に、前記コード・ブロックの前記最上位ビットプレーン内の全てのシンボルをエントロピ符号化することを特徴とする装置。

【請求項２６】　請求項２５に記載の装置において、
前記リスト・マネージャは、
前記第１のビットプレーン用として、前記シグニフィカンス・プロパゲーション・パス中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の前記第１のリストを前記最初のクリーンアップ・ステップ・パス中に生成する手段と、
前記第１のビットプレーン用の前記マグニチュード・リファインメント・パス中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の前記第２のリストを、前記最初のクリーンアップ・ステップ・パス中生成する手段と、を有することを特徴とする装置。

【請求項２７】　請求項２５に記載の装置において、
前記リスト・マネージャは、
前記最初のクリーンアップ・ステップ符号化パス中、符号化された前記各シンボルについて、前記エントロピ符号化されたシンボルが属する前記係数の前記有意状態を決定する手段を有し、前記決定手段は、
前記有意状態が１であれば、第１のフラグを１にセットする手段と、
前記有意状態が１であれば、前記所定の１の有意状態を有する前記係数の前記近傍係数を決定し、前記各近傍係数について第２のフラグを１にセットする手段と、
前記所定の有意状態がゼロであれば、前記第１のフラグをゼロにセットする手段と、を有するように構成される決定手段と、
前記第１のビットプレーン用の前記マグニチュード・リファインメント・パス中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の前記第２のリストを前記最初のクリーンアップ・ステップ符号化パス中生成する手段を有し、前記位置に関連する前記第１のフラグが１の場合、前記係数の前記位置が前記第２のリストに追加されるように構成される手段と、
前記第１のビットプレーンについて前記シグニフィカンス・プロパゲーション・パス中、エントロピ符号化される前記シンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の前記第１のリストを生成する手段を有し、前記位置に関連する前記第２のフラグが１であり、かつ前記位置に関連する前記第１のフラグがゼロである場合、前記係数の前記位置を前記第１のリストに追加することを特徴とする装置。

【請求項２８】　請求項１６に記載の装置において、エントロピ符号化の前記装置が前記シンボルのエントロピ符号化または復号を行うことが可能であることを特徴とする装置。

【請求項２９】　請求項１６に記載の装置において、エントロピ符号化の前記装置が前記シンボルのエントロピ符号化装置であり、前記リスト・メモリ・マネージャが現在のビットプレーン用の前記第１、第２、第３のリストを現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パスに先行して生成することを特徴とする装置。

【請求項３０】　請求項２９に記載の装置において、
前記リスト・メモリ・マネージャは、現在のビットプレーン番号Ｎ以上の有意状態を持つ前記第２のリストに前記コード・ブロック内の係数の前記位置を追加する手段を有することを特徴とする装置。

【請求項３１】　請求項２９に記載の装置において、前記リスト・メモリ・マネージャは、現在のビットプレーン番号Ｎ未満の有意状態を持ち、かつ、所定の走査順でＮ以上の有意状態を持つ係数である、前記コード・ブロック内の任意の係数の前記位置を近傍係数に先行する前

記第１のリストに追加する手段を有することを特徴とする装置。

【請求項３２】　デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化するコンピュータ・プログラムを有するコンピュータ・プログラム製品であって、

前記コンピュータ・プログラムは、

第１のビットプレーンから所定の最下位ビットプレーンへシンボルから成るビットプレーンに前記コード・ブロックを分割するためのコードと、

シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中に前記シンボルをエントロピ符号化するためのコードと、

現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第１のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パスに先行して生成するためのコードと、

現在のビットプレーンのマグニチュード・リファインメント・パス中、エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第２のリストを、現在のビットプレーンの前記マグニチュード・リファインメント・パスに先行して生成するためのコードと、

現在のビットプレーンのクリーンアップ・パス中、エントロピ符号化の対象となるシンボルを有する、前記コード・ブロック内の前記係数の位置の第３のリストを、現在のビットプレーンの前記クリーンアップ・パスに先行して生成するためのコードと、

前記第１の位置リスト、第２の位置リスト、第３の位置リストとを格納するためのコードと、を有することを特徴とするコンピュータ・プログラム製品。

【請求項３３】　係数を有するブロックを表すシンボルのエントロピ符号化方法であって、

第１のビットプレーンから所定の最小ビットプレーンまでの各ビットプレーンについて、

前記ブロックの現在のビットプレーン中、エントロピ符号化されるシンボルを有する、ブロック内の前記係数の位置の少なくとも１つのリストを生成するステップと、

前記少なくとも１つのリストを用いて、前記ブロックの現在のビットプレーン内の係数の前記シンボルをエントロピ符号化するステップと、を実行することを特徴とする方法。

【請求項３４】　係数を有するブロックを表すシンボルをエントロピ符号化する装置であって、

前記ブロックの現在のビットプレーン中、エントロピ符号化されるシンボルを有する、前記ブロック内の前記係数の位置の少なくとも１つのリストを生成する生成ユニットと、

前記少なくとも１つのリストを用いて、前記ブロックの現在のビットプレーン内の係数の前記シンボルをエントロピ符号化するエントロピ符号器と、を有することを特徴とする装置。

【請求項３５】　係数を有するブロックを表すシンボルをエントロピ符号化するコンピュータ・プログラムを有するコンピュータ・プログラム製品であって、

前記ブロックの現在のビットプレーン中、エントロピ符号化されるシンボルを有する、前記ブロック内の前記係数の位置の少なくとも１つのリストを生成するためのコードと、

前記少なくとも１つのリストを用いて、前記ブロックの現在のビットプレーン内の係数の前記シンボルをエントロピ符号化するためのコードと、を有することを特徴とするコンピュータ・プログラム製品。

【請求項３６】　デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する方法であって、

第１のビットプレーンから所定の最下位ビットプレーンまでの各ビットプレーンについて、

現在のビットプレーン内でエントロピ符号化の対象となる前記シンボルをそれぞれ持つ係数の、第１、第２及び第３の位置リストに従って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中に、前記コード・ブロックの現在のビットプレーンの前記変換係数のシンボルを符号化するステップと、

現在のビットプレーンの後の次のビットプレーン用の位置の第１のリストと、現在のビットプレーンの後の次のビットプレーン用の位置の第２のリストと、現在のビットプレーン用の位置の第３のリストとを生成するステップと、を有し、

前記生成するステップは、

ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に生成するサブステップと、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数である、現在のビットプレーン内の係数の位置の第１のリストの位置に、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パスでタグを付けるサブステップと、

現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数の前

記位置の第４のリストに、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中タグを付けるサブステップと、
現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に現在のビットプレーン用の位置の前記第３のリストを生成するサブステップであって、前記第３のリストが、現在のビットプレーン用の位置の前記第２のリスト内の位置と、前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されたシンボルを持つ係数の位置と、を除くコード・ブロックの全ての位置リストとを含むように構成されるサブステップと、
現在のビットプレーンの前記クリーンアップ・パス中エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンのクリーンアップ・パス中に１の有意状態に変化する係数である、コード・ブロック内の前記係数の位置の第５のリストを、現在のビットプレーンのクリーンアップ・パス中に生成するサブステップと、
現在のビットプレーンのクリーンアップ・パス中にエントロピ符号化されたシンボルを持ち、現在のビットプレーンのクリーンアップ・パス中にゼロの有意状態を持つ係数である、コード・ブロック内の前記係数の位置の第６のリストを、現在のビットプレーンの前記クリーンアップ・パス中に生成するサブステップと、
現在のビットプレーンの前記クリーンアップ・パス中に前記タグを付けた第１の位置リストと、タグを付けなかった第１の位置リストとを現在のビットプレーンについてソートして、それぞれ第７の位置リストと、第８の位置リストとに変えるサブステップと、
現在のビットプレーンの前記クリーンアップ・パス中に前記タグを付けた第４の位置リストと、タグを付けなかった第４の位置リストとを現在のビットプレーンについてソートして、それぞれ第９の位置リストと、第１０の位置リストとに変えるサブステップと、
現在のビットプレーンのクリーンアップ・パス中に前記第２、第５、第７及び第９のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成するサブステップと、
現在のビットプレーンのクリーンアップ・パス中に前記第６、第８及び第１０のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストを生成するサブステップと、
現在のビットプレーンのクリーンアップ・パス中に次のビットプレーンのための前記生成された第１のリストから、１の有意状態を持つ係数である、コード・ブロック内の任意の係数の位置を取り除くサブステップと、を有することを特徴とする方法。

【請求項３７】　デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する装置において、
現在のビットプレーン内でエントロピ符号化されたシンボルをそれぞれ持つ、第１と、第２と、第３の係数の位置リストに従って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中にコード・ブロックの現在のビットプレーンの前記変換係数の前記シンボルを符号化するエントロピ符号器と、
現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストと、現在のビットプレーンの後の次のビットプレーン用の位置の第２のリストと、現在のビットプレーン用の位置の第３のリストとを生成するリスト・メモリ・マネージャと、を有し、
前記リスト・メモリ・マネージャは、
ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中生成する手段と、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数である、現在のビットプレーン内の位置の係数の第１のリストの位置に、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中タグを付ける手段と、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数の前記位置の第４のリストに、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中タグを付ける手段と、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に現在のビットプレーン用の位置の前記第３のリストを生成する手段であって、前記第３のリストが、現在のビットプレーン用の位置の前記第２のリスト内の位置と、前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されたシンボルを持つ係数の位置と、を除くコード・ブロックのすべての位置リストとを含むように構成される手段と、
現在のビットプレーンの前記クリーンアップ・パス中エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンの前記クリーンアップ・パス中に１の有意状態に変化する係数である、前記コード・ブロック内の前記係数の位置の第５のリストを、現在のビットプレーンの前記クリーンアップ・パス中生成する手段と、
現在のビットプレーンの前記クリーンアップ・パス中エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンの前記クリーンアップ・パス中ゼロの有意状態を持つ係数である、前記コード・ブロック内

の前記係数の位置の第６のリストを、現在のビットプレーンの前記クリーンアップ・パス中生成する手段と、
現在のビットプレーンの前記クリーンアップ・パス中に前記タグを付けた第１の位置リストと、タグを付けなかった第１の位置リストとを現在のビットプレーンについてソートして、それぞれ第７の位置リストと、第８の位置リストとに変える手段と、
現在のビットプレーンの前記クリーンアップ・パス中に前記タグを付けた第４の位置リストと、タグを付けなかった第４の位置リストとを現在のビットプレーンについてソートして、それぞれ第９の位置リストと、第１０の位置リストとに変える手段と、
現在のビットプレーンの前記クリーンアップ・パス中に前記第２、第５、第７及び第９のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成する手段と、
現在のビットプレーンの前記クリーンアップ・パス中に前記第６、第８及び第１０のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストを生成する手段と、
現在のビットプレーンの前記クリーンアップ・パス中に次のビットプレーンのための前記生成された第１のリストから、１の有意状態を持つ係数である、前記コード・ブロック内の任意の係数の位置を取り除く手段と、を有することを特徴とする装置。
【請求項３８】　デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化するコンピュータ・プログラムを有するコンピュータ・プログラム製品であって、
前記コンピュータ・プログラムは、
現在のビットプレーン内でそれぞれエントロピ符号化されたシンボルを持つ係数の第１と、第２と、第３の位置リストに従って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中にコード・ブロックの現在のビットプレーンの前記変換係数の前記シンボルをエントロピ符号化するためのコードと、
現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストと、現在のビットプレーンの後の次のビットプレーン用の位置の第２のリストと、現在のビットプレーン用の位置の第３のリストとを生成するためのコードと、を有し、
前記生成するための用コードは、
ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中生成するコードと、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数である、現在のビットプレーン内の係数の位置の第１のリストの位置に、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中タグを付けるコードと、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数の前記位置の第４のリストの位置に対して、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中タグを付けるコードと、
現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中に現在のビットプレーン用の位置の前記第３のリストを生成するコードであって、前記第３のリストが、現在のビットプレーン用の位置の前記第２のリスト内の位置と、前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されたシンボルを持つ係数の位置と、を除くコード・ブロックの全ての位置リストとを含むように構成されるコードと、
現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持ち、さらに、現在のビットプレーンのクリーンアップ・パス中に１の有意状態に変化するコード・ブロック内の前記係数の位置の第５のリストを、現在のビットプレーンの前記クリーンアップ・パス中生成するコードと、
現在のビットプレーンの前記クリーンアップ・パス中エントロピ符号化されたシンボルを持ち、現在のビットプレーンの前記クリーンアップ・パス中ゼロの有意状態を持つ係数である、コード・ブロック内の前記係数の位置の第６のリストを、現在のビットプレーンの前記クリーンアップ・パス中に生成するコードと、
現在のビットプレーンの前記クリーンアップ・パス中に前記タグを付けた第１の位置リストと、タグを付けなかった第１の位置リストとを現在のビットプレーンについてソートして、それぞれ第７の位置リストと、第８の位置リストとに変えるコードと、
現在のビットプレーンの前記クリーンアップ・パス中に前記タグを付けた第４の位置リストと、タグを付けなかった第４の位置リストとを現在のビットプレーンについてソートして、それぞれ第９の位置リストと、第１０の位置リストとに変えるコードと、
現在のビットプレーンの前記クリーンアップ・パス中に前記第２、第５、第７及び第９のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成するコードと、
現在のビットプレーンの前記クリーンアップ・パス中に前記第６、第８及び第１０のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプ

レーン用の位置の前記第１のリストを生成するコードと、
現在のビットプレーンの前記クリーンアップ・パス中に次のビットプレーンのための前記生成された第１のリストから、１の有意状態を持つ係数である、前記コード・ブロック内の任意の係数の位置を取り除くコードと、を有することを特徴とするコンピュータ・プログラム製品。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は一般にデータ圧縮と解凍に関し、特に、エントロピ符号化及び復号化方法及びその装置に関する。本発明はまた、エントロピ符号化を行うためのコンピュータ・プログラムを有するコンピュータ・プログラム製品に関するものである。

【０００２】

【従来の技術】新しいＪＰＥＧ-２０００規格に対する提案要求が最近出され、“情報技術－ＪＰＥＧ２０００画像符号化システム－ＪＰＥＧ２０００委員会草案バージョン１.０（１９９９年１２月９日）という表題の規格草案（本明細書では以後ＪＰＥＧ２０００と呼ぶ）が発表された。

【０００３】ＪＰＥＧ２０００では１以上の画像タイル成分に画像全体を分割し、次いで、この画像タイル成分の各々の２次元離散ウェーブレット変換を行うことが提案されている。次いで、各画像タイル成分の変換係数がサブバンドにグループ化される。次いで、各コード・ブロックがエントロピ符号化される前に、該サブバンドは分割されて矩形のコード・ブロックに変えられる。各コード・ブロックの変換係数はエントロピ符号化に先行して符号絶対値で表現される。エントロピ符号化は、コンテキスト生成部と算術符号化部の２つの部分から構成される。算術符号化部は、符号化の対象となる係数のビット・シンボルと、そのビット・シンボルのコンテキストとを入力としている。ある係数のビット・シンボルのコンテキストは、コード・ブロックの同じビットプレーン内の８個の包囲係数の‘有意（significance）’状態に基づいている。該ビット・シンボルは算術符号化部によって符号化される。係数の‘有意’状態とは、２進値変数Ｓi[m，n]であり、該２進値変数は０に初期化されるが、係数の第１のノンゼロ・ビット値が符号化された直後にこの変数は１に変化する。図１は、係数“Ｘ[m，n]”の８個の包囲係数の近傍有意状態Ｓi[m－１，n－１]、Ｓi[m－１，n]、Ｓi[m－１，n＋１]、Ｓi[m，n－１]、Ｓi[m，n＋１]、Ｓi[m＋１，n－１]、Ｓi[m＋１，n]とＳi[m＋１，n＋１]を示す。但し、m、nはそれぞれコード・ブロックの行番号と列番号であり、Ｓi []は、係数Ｘ[m，n]のビット・シンボルｉが符号化される直前の有意状態である。これらの近傍有意状態は係数Ｘ[m，n]の３×３近傍有意状態と呼ばれることもある。

【０００４】算術符号化部[m，n]は、１つのコード・ブロックの最上位ビットプレーンの全てのビット・シンボルをまず符号化し、次いで、コード・ブロックの次に低いビットプレーンの全てのビット・シンボル等々を符号化し、最後に最下位ビットプレーンの全てのビット・シンボルを符号化する。１つのコード・ブロックの各ビットプレーンの範囲内で、算術符号化部は所定の順序の３つのパスで係数のビット・シンボルを符号化する。

【０００５】ビットプレーンの第１のパスはシグニフィカンス・プロパゲーション・パス（ＳＰパス）と呼ばれ、ビットプレーンの第２のパスはマグニチュード・リファインメント・パス（ＭＲパス）呼ばれ、ビットプレーンの第３で最後のパスはクリーンアップ・パス（Ｎパス）と呼ばれる。ＳＰパス中で、符号化対象ビット・シンボルは１の有意状態を持っているが、該ビット・シンボル自身が０の有意状態を持つ近傍ビット・シンボルを持っている場合、ビットプレーンのビット・シンボルは符号化される。ＭＲパス中で、ビットプレーンのビット・シンボルがまだ符号化されていない場合、該ビット・シンボルの係数が前回の符号化ビットプレーンですでに有意であれば、該ビット・シンボルは符号化される。Ｎパス中で、予め符号化されなかったビットプレーンの残りのビット・シンボルはこの時符号化される。

【０００６】このコンテキストは符号化対象ビットと共に算術符号化部へ転送され、符号化されたシンボルはビット・ストリームへ出力される。符号化対象のこのビット・シンボルの値が１で、かつ、その有意状態がゼロであれば、いったんビット・シンボルが符号化され、かつ、直後の符号化対象ビット・シンボルが係数用の符号ビットであった場合、有意状態は１にセットされる。上記条件を満たさない場合には有意状態はゼロ（０）のままである。連続する係数とパスのコンテキストを考慮する場合、この係数の最も現在の有意状態が用いられる。

【０００７】算術符号化部は、所定の同じ順序で３つのパス（ＳＰ、ＭＲ、Ｎ）内のビットプレーンのビット・シンボルを符号化する。算術符号化部はまず中にノンゼロ・ビットを持つ最上位ビットプレーンへ進み、ＳＰ、ＭＲパスをスキップし、Ｎパスから始まる。次いで、算術符号化部は次に低いビットプレーンへ進み、３つのパス（ＳＰ、ＭＲ、Ｎ）で、その順序でビット・シンボルを符号化する。次いで、算術符号化部は次に低いビットプレーンへ進み、同順のパス（ＳＰ、ＭＲ、Ｎ）でビット・シンボル等々を符号化し、最後に最下位ビットプレーンを符号化する。

【０００８】さらに、コード・ブロックの各ビットプレーンは特別の順序で走査される。最上部左側から始まり、列の最初の４個のビット・シンボルが走査される。次いで、コード・ブロックの幅がカバーされてしまうまで、第２列の最初の４個のビット・シンボルが走査され

る。次いで、第１列の第２番目の４個のビット・シンボル等々と走査される。任意の残りの行について、サブバンドの最下位コード・ブロック内で同様の走査が続けられる。図２は、８×８ブロックで構成される６４個の変換係数を持つコード・ブロックについてのコード・ブロック走査パターンの一例を示す。この例でわかるように、走査は４つの行の連続するストリップ（細長い片）で行われる。コード・ブロックは８×８ブロックに限定されるものではなく、６４×６４のコード・ブロック等のさらに大きなコード・ブロックが考えられる。後者の場合、４つの行の１６個の連続するストリップが生じる。説明を簡単にするために、本明細書ではこの走査順をＪＰＥＧ２０００走査順と呼ぶことにする。

【０００９】ＪＰＥＧ２０００に記述されているエントロピ復号化は、エントロピ符号化と鏡像の関係になる。例えば、デコーダは符号化の場合と同順でシンボルを復号する。エントロピ復号化はまた、コンテキスト生成部と算術復号部の２つのセクションから構成される。算術復号化部は、復号の対象となるシンボルと、復号の対象となるそのシンボルのコンテキストとを入力として受け取る。復号の対象となる走査順序シンボルのコンテキストは、コード・ブロックの同じビットプレーン内の８個の包囲係数の‘有意’状態に基づき、該シンボルは算術復号化部によって復号される。係数の‘有意’状態は２進値変数Ｓ[m，n]であり、該２進値変数は０に初期化されるが、係数の第１のノンゼロ・ビットプレーン値が復号されると１に変化する。このように、復号化段階中の係数の有意状態は符号化段階中の係数の有意状態と鏡像関係になる（図２参照）。

【００１０】ＪＰＥＧ２０００では、算術符号化と復号化とは最上位ビットプレーンから最下位ビットプレーンへビットプレーン毎に行われる。初めに、符号化／復号化は、その中にゼロしか含まないすべてのビットプレーンをスキップし、その中にノンゼロ要素を持つ第１のビットプレーンに対するオペレーションを開始する。いったん符号化／復号化が、その中にノンゼロ要素を持つ第１のビットプレーンに達すると、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パスの３つのパスで、コーデックは各連続ビットプレーンに対して作用する。

【００１１】ビットプレーン全体にわたるコーデック走査を有し、或る特徴を持つ位置の配置を定めるＪＰＥＧ２０００を実現するエントロピ・コーデックが提案されている。例えば、シグニフィカンス・プロパゲーション・パスで、コーデックは、位置自体は有意でないが、その位置の３×３の近傍に少なくとも１つの有意係数を持つ位置を探す。マグニチュード・リファインメント・パスで、コーデックは既に有意な位置を探す。クリーンアップ・パスで、コーデックは前に符号化／復号化されていない位置を探す。したがって、この提案されたコーデックでは、全てのビットプレーンについて３回の走査が行われる。これは、最大スループットが３サイクル当たり１シンボルであることを含意する。

【００１２】上記から解るように、提案されたコーデックの主要な問題点として、各パスについて、コーデックが全ての位置を走査し、現在のパスで符号化／復号化が行われているかどうかを判定する必要があるという点が挙げられる。これは、符号化の必要がない位置を走査してサイクルが浪費されることを意味する。

【００１３】

【発明が解決しようとする課題】本構成及び／又は提案の１以上の問題点を実質的に解決または少なくとも改善することが本発明の目的である。

【００１４】

【課題を解決するための手段】本発明の第１の態様によれば、デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する方法が提供される。該方法は第１のビットプレーンから所定の最下位ビットプレーンまでの各ビットプレーンについて以下のステップ：現在のビットプレーン内にゼロの有意状態を持つ係数であって、現在のビットプレーン内に１の有意状態を持つ近傍係数を持つ係数に属するシンボルである、現在のビットプレーン内のすべてのシンボルのエントロピ符号化を行うシグニフィカンス・プロパゲーション・ステップと、；前回のビットプレーン内に１の有意状態を持つ係数に属するシンボルである、現在のビットプレーン内のすべてのシンボルのエントロピ符号化を行うマグニチュード・リファインメント・ステップと、；現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップまたはマグニチュード・リファインメント・ステップ中、それ以前にエントロピ符号化されなかったシンボルである、現在のビットプレーン内のすべてのシンボルのエントロピ符号化を行うクリーンアップ・ステップと、を実行する方法において、現在のビットプレーンのシグニフィカンス・プロパゲーション・ステップ中エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第１のリストを、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・ステップに先行して生成するステップと、；現在のビットプレーンのマグニチュード・リファインメント・ステップ中エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第２のリストを、現在のビットプレーンのマグニチュード・リファインメント・ステップに先行して生成するステップと、；現在のビットプレーンの前記クリーンアップ・ステップ中エントロピ符号化の対象となるシンボルを持つ係数である、前記コード・ブロック内の前記係数の位置の第３のリストを、現在のビットプレーンのクリーンアップ・ステッ

プに先行して生成するステップとをさらに有する。

【００１５】本発明の別の態様によれば、デジタル画像の変換係数を有する、コード・ブロックを表すシンボルをエントロピ符号化する装置が提供される。該装置は、第１のビットプレーンから所定の最下位ビットプレーンまで前記コード・ブロックをシンボルから成るビットプレーンに分割するビットプレーン・スプリッタと；シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中に前記シンボルを符号化するエントロピ符号器と；現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第１のリストを、現在のビットプレーンのシグニフィカンス・プロパゲーション・パスに先行して生成し、現在のビットプレーンのマグニチュード・リファインメント・パス中エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第２のリストを、現在のビットプレーンのマグニチュード・リファインメント・パスに先行して生成し、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第３のリストを、現在のビットプレーンのクリーンアップ・パスに先行して生成するリスト・メモリ・マネージャと；前記第１のリストと、第２のリストと、第３の位置リストとを格納する記憶装置と、を有する。

【００１６】本発明の別の態様によれば、デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化するコンピュータ・プログラムを有するコンピュータ・プログラム製品が提供される。該コンピュータ・プログラムは、第１のビットプレーンから所定の最下位ビットプレーンへシンボルから成るビットプレーンに前記コード・ブロックを分割するためのコードと、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中に前記シンボルをエントロピ符号化するためのコードと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第１のリストを、現在のビットプレーンのシグニフィカンス・プロパゲーション・パスに先行して生成するためのコードと、現在のビットプレーンのマグニチュード・リファインメント・パス中エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第２のリストを、現在のビットプレーンのマグニチュード・リファインメント・パスに先行して生成するためのコードと、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化の対象となるシンボルを持つ係数である、コード・ブロック内の前記係数の位置の第３のリストを、現在のビットプレーンのクリーンアップ・パスに先行して生成するためのコードと、前記第１の位置リストと、第２の位置リストと、第３の位置リストとを格納するためのコードと、を有する。

【００１７】本発明の別の態様によれば、係数を有するブロックを表すエントロピ符号化シンボルの方法が提供される。該方法は、第１のビットプレーンから所定の最下位ビットプレーンまでの各ビットプレーンについて以下のステップ：ブロックの現在のビットプレーン中エントロピ符号化されるシンボルを有する係数である、ブロック内の前記係数の位置の少なくとも１つのリストを生成するステップと、前記少なくとも１つのリストを用いて、そのブロックの現在のビットプレーン内の係数の前記シンボルをエントロピ符号化するステップと、を実行する。

【００１８】本発明の別の態様によれば、係数を有するブロックを表すシンボルをエントロピ符号化する装置が提供される。該装置は、ブロックの現在のビットプレーン中エントロピ符号化されるシンボルを持つ係数である、ブロック内の前記係数の位置の少なくとも１つのリストを生成する生成装置と、前記少なくとも１つのリストを用いて、そのブロックの現在のビットプレーン内の係数の前記シンボルをエントロピ符号化するエントロピ符号器と、を有する。

【００１９】本発明の別の態様によれば、係数を有するブロックを表すシンボルをエントロピ符号化するコンピュータ・プログラムを有するコンピュータ・プログラム製品が提供される。該コンピュータ・プログラムは、ブロックの現在のビットプレーン中エントロピ符号化されるシンボルを有する係数である、ブロック内の前記係数の位置の少なくとも１つのリストを生成するためのコードと、前記少なくとも１つのリストを用いて、そのブロックの現在のビットプレーン内の係数の前記シンボルをエントロピ符号化するためのコードと、を有する。

【００２０】本発明の別の態様によれば、デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する方法が提供される。該方法は、第１のビットプレーンから所定の最下位ビットプレーンまでの各ビットプレーンについて以下のステップ：現在のビットプレーン内でエントロピ符号化の対象となる前記シンボルをそれぞれ持つ係数の、第１と、第２と、第３の位置リストに従って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中にコード・ブロックの現在のビットプレーンの前記変換係数のシンボルを符号化するステップと、現在のビットプレーンの後の次のビットプレーン用の位置の第１のリストと、現在のビットプレーンの後の次のビットプレーン用の位置の第２のリストと、現在のビットプレーン用の位置の第３のリスト

とを生成するステップと、が提供される。この場合、該生成ステップは、ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中生成するサブステップと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数である、現在のビットプレーン内の係数の位置の第１のリストの位置に現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中タグを付けるサブステップと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数の前記位置の第４のリストに現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中タグを付けるサブステップと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に現在のビットプレーン用の位置の前記第３のリストを生成するサブステップであって、前記第３のリストが、現在のビットプレーン用の位置の前記第２のリスト内の位置と、シグニフィカンス・プロパゲーション・パス中エントロピ符号化されたシンボルを持つ係数の位置と、を除くコード・ブロックのすべての位置リストとを含むように構成されるサブステップと、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンのクリーンアップ・パス中に１の有意状態に変化する係数である、コード・ブロック内の前記係数の位置の第５のリストを現在のビットプレーンのクリーンアップ・パス中生成するサブステップと、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持ち、現在のビットプレーンのクリーンアップ・パス中ゼロの有意状態を持つ係数である、コード・ブロック内の前記係数の位置の第６のリストを現在のビットプレーンのクリーンアップ・パス中生成するサブステップと、現在のビットプレーンのクリーンアップ・パス中に前記タグを付けた第１の位置リストと、タグを付けなかった第１の位置リストとを現在のビットプレーンについてソートして、それぞれ第７の位置リストと、第８の位置リストとに変えるサブステップと、現在のビットプレーンのクリーンアップ・パス中に前記タグを付けた第４の位置リストと、タグを付けなかった第４の位置リストとを現在のビットプレーンについてソートして、それぞれ第９の位置リストと、第１０の位置リストとに変えるサブステップと、現在のビットプレーンのクリーンアップ・パス中に前記第２、第５、第７及び第９のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーンの位置の前記第２のリストを生成するサブステップと、現在のビットプレーンのクリーンアップ・パス中に前記第６、第８及び第１０のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーンの位置の前記第１のリストを生成するサブステップと、現在のビットプレーンのクリーンアップ・パス中に次のビットプレーンのための前記生成された第１のリストから、１の有意状態を持つ係数である、コード・ブロック内の任意の係数の位置を取り除くサブステップと、を有する。

【００２１】本発明の別の態様によれば、デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化する装置が提供される。該装置は、現在のビットプレーン内でエントロピ符号化されたシンボルをそれぞれ持つ、第１と、第２と、第３の係数の位置リストに従って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中にコード・ブロックの現在のビットプレーンの前記変換係数の前記シンボルを符号化するエントロピ符号器と、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストと、現在のビットプレーンの後の次のビットプレーン用の位置の第２のリストと、現在のビットプレーン用の位置の第３のリストとを生成するリスト・メモリ・マネージャと、を有し、リスト・メモリ・マネージャが、ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中生成する手段と、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数である、現在のビットプレーン内の係数の位置の第１のリストの位置に現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中タグを付ける手段と、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数の前記位置の第４のリストに現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中タグを付ける手段と、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に現在のビットプレーン用の位置の前記第３のリストを生成する手段であって、前記第３のリストが、現在のビットプレーン用の位置の前記第２のリスト内の位置と、シグニフィカンス・プロパゲーション・パス中エントロピ符号化されたシンボルを持つ係数の位置と、を除くコード・ブロックのすべての位置リストとを含むように構成される手段と、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンのクリーンアップ・パス中に１の有

意状態に変化する係数である、コード・ブロック内の前記係数の位置の第５のリストを現在のビットプレーンのクリーンアップ・パス中生成する手段と、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持つ係数であって、現在のビットプレーンのクリーンアップ・パス中ゼロの有意状態を持つ係数である、コード・ブロック内の前記係数の位置の第６のリストを現在のビットプレーンのクリーンアップ・パス中生成する手段と、現在のビットプレーンのクリーンアップ・パス中に前記タグを付けた第１の位置リストと、タグを付けなかった第１の位置リストとを現在のビットプレーンについてソートして、それぞれ第７の位置リストと、第８の位置リストとに変える手段と、現在のビットプレーンのクリーンアップ・パス中に前記タグを付けた第４の位置リストと、タグを付けなかった第４の位置リストとを現在のビットプレーンについてソートして、それぞれ第９の位置リストと、第１０の位置リストとに変える手段と、現在のビットプレーンのクリーンアップ・パス中に前記第２、第５、第７及び第９のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成する手段と、現在のビットプレーンのクリーンアップ・パス中に前記第６、第８及び第１０のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストを生成する手段と、現在のビットプレーンのクリーンアップ・パス中に次のビットプレーンのための前記生成された第１のリストから、１の有意状態を持つ係数である、コード・ブロック内の任意の係数の位置を取り除く手段と、を有する。

【００２２】本発明の別の態様によれば、デジタル画像の変換係数を有するコード・ブロックを表すシンボルをエントロピ符号化するコンピュータ・プログラムを有するコンピュータ・プログラム製品が提供される。該コンピュータ・プログラムは、現在のビットプレーン内でそれぞれエントロピ符号化されたシンボルを持つ係数の第１と、第２と、第３の位置リストに従って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パス中にコード・ブロックの現在のビットプレーンの前記変換係数の前記シンボルをエントロピ符号化するためのコードと、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストと、現在のビットプレーンの後の次のビットプレーン用の位置の第２のリストと、現在のビットプレーン用の位置の第３のリストとを生成するためのコードと、を有し、その場合、該生成用コードは、ゼロの非有意状態を持つ係数であって、現在のビットプレーンの前記シグニフィカンス・プロパゲーション・パス中エントロピ符号化されるシンボルである、１の有意状態を持つシンボルを持つ係数に隣接する前記係数である、前記コード・ブロック内の前記係数の位置の第４のリストを現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中生成するコードと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数である、現在のビットプレーン内の係数の位置の第１のリストの位置に現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中タグを付けるコードと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に１の有意状態に変化する係数の前記位置の第４のリストの位置に対して現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中タグを付けるコードと、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス中に現在のビットプレーン用の位置の前記第３のリストを生成するコードであって、前記第３のリストが、現在のビットプレーン用の位置の前記第２のリスト内の位置と、シグニフィカンス・プロパゲーション・パス中エントロピ符号化されたシンボルを持つ係数の位置と、を除くコード・ブロックのすべての位置リストとを含むように構成されるコードと、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持ち、さらに、現在のビットプレーンのクリーンアップ・パス中に１の有意状態に変化するコード・ブロック内の前記係数の位置の第５のリストを現在のビットプレーンのクリーンアップ・パス中生成するコードと、現在のビットプレーンのクリーンアップ・パス中エントロピ符号化されたシンボルを持ち、現在のビットプレーンのクリーンアップ・パス中ゼロの有意状態を持つ係数である、コード・ブロック内の前記係数の位置の第６のリストを現在のビットプレーンのクリーンアップ・パス中生成するコードと、現在のビットプレーンのクリーンアップ・パス中に前記タグを付けた第１の位置リストと、タグを付けなかった第１の位置リストとを現在のビットプレーンについてソートして、それぞれ第７の位置リストと、第８の位置リストとに変えるコードと、現在のビットプレーンのクリーンアップ・パス中に前記タグを付けた第４の位置リストと、タグを付けなかった第４の位置リストとを現在のビットプレーンについてソートして、それぞれ第９の位置リストと、第１０の位置リストとに変えるコードと、現在のビットプレーンのクリーンアップ・パス中に前記第２、第５、第７及び第９のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第２のリストを生成するコードと、現在のビットプレーンのクリーンアップ・パス中に前記第６、第８及び第１０のリスト上の前記位置を比較して併合し、現在のビットプレーンの後の次のビットプレーン用の位置の前記第１のリストを生成するコードと、現在のビットプレーンのクリーンアップ・パス中に次のビットプレーンのための前記生成された第１のリストから、１の有意状態を持

つ係数である、コード・ブロック内の任意の係数の位置を取り除くコードと、を有する。

【００２３】

【発明の実施の形態】本発明のいくつかの好ましい実施の形態について、以下に図面を参照しながら説明する。

【００２４】ステップ及び／又は特徴を示す同じ参照番号を持つ１以上のいずれかの添付図面で参照が行われる場合、それらのステップ及び／又は特徴は、逆の意図が現れない限り、この説明の目的のために同じ機能または作用を持つ。

【００２５】本明細書に記載の好ましい装置と方法の原理は、データ圧縮と解凍に対して一般的適用性を持つ。しかし、説明を簡単にするために、上述のＪＰＥＧ２０００に従うデジタル画像の圧縮と解凍を参照しながら好ましい装置と方法について説明する。しかし、これは記載の方法及び装置に本発明を限定することを意図するものではない。例えば、本発明はデジタル・ビデオの圧縮と解凍に適用することも可能である。

【００２６】好ましい方法についての詳細な説明に進む前に、好ましい方法の簡単な概観について以下説明する。まず、画像全体は１以上の画像タイル成分に分割され、次いで、これら画像タイル成分の各々に対して２次元離散ウェーブレット変換が行われる。次いで、各画像タイル成分の変換係数がサブバンドにグループ化される。次いで、該サブバンドは、各コード・ブロックがエントロピ符号化される前にさらに矩形のコード・ブロックに分割される。

【００２７】次に図３に目を向けると、第１の構成に従ってコード・ブロックの変換係数のシンボルをエントロピ符号化する方法を示すフローチャート３００が示されている。好適には、方法３００が画像の各コード・ブロックを呼び出すことが望ましい。本方法はコーデック法で機能する。すなわち、本方法はシンボルを復号もしくは符号化することができる。符号化あるいは復号のいずれかが行われる間、本方法はほぼ同一のステップを実行する。復号処理は、シグニフィカンス・プロパゲーションリストと、マグニチュード・リファインメント・リストと、クリーンアップ・リストとを利用するという点で符号化処理と鏡像関係になる。しかし、復号化中、ビットプレーンに分割されるソースコード・ブロックは存在せず、最上位ビットプレーンのロケーションが既に与えられている。説明を簡単にするために、符号化処理を参照しながら主としてエントロピ符号化法について説明を行う。

【００２８】方法３００は、ステップ３０２から始まり、いずれかの必要なパラメータが初期化される。初期化段階中、方法３００は有効値テーブルを生成する。この有効値テーブルは符号化の対象となるコード・ブロックの係数の有意状態を示す２次元配列を有する。最初に、配列の各エントリは、各係数が非有意状態を持っていることを示すゼロ（０）にセットされる。ある係数の第１のノンゼロ・ビットプレーン値が符号化され、配列の対応するエントリが（１）にリセットされたとき、その係数の有意状態は（１）に遷移する。任意の呼び出されたシグニフィカンス・プロパゲーション符号化パス中にまたは、任意のクリーンアップ符号化パス中に有効値テーブルの更新を行うことができる。

【００２９】また初期化段階中、本方法によって、本明細書でシグニフィカンス・プロパゲーションリスト（ＳＰリスト）、マグニチュード・リファインメント・リスト（ＭＲリスト）及びクリーンアップ・リスト（Ｎリスト）と名付けられる３つのリストが生成される。これらのリストには、コード・ブロックの範囲内の、現在のパス中符号化の対象となるビット・シンボルの位置が含まれる。例えば、シグニフィカンス・プロパゲーションリストには、コード・ブロックの範囲内に、現在のシグニフィカンス・プロパゲーション・パス中符号化の対象となるビット・シンボルのすべての位置リストが含まれる。ビット・シンボルの位置という用語は、本明細書では、そのビット・シンボルが一部を形成する変換係数のコード・ブロックの範囲内の配列位置を意味する。さらに、これらのリストの各々の範囲内の位置は、コード・ブロック走査順に従って索引化される（図２参照）。最初、シグニフィカンス・プロパゲーションリスト（ＳＰリスト）と最大リファインメント・リスト（ＭＲリスト）とは空である。一方、クリーンアップ・リスト（Ｎリスト）はコード・ブロックの範囲内のすべての位置をリストし、ノンゼロ・ビットを持つ最上位のビットプレーンの符号化に使用される。

【００３０】ＳＰリストとＭＲリストとは、必要に応じてさらなる位置の追加と削除とにより本方法の処理中更新することができる。さらに、現在のＮリストは現在のＳＰとＭＲリスト内に含まれていない位置を含むようにＳＰ符号化パス中生成される。

【００３１】開始３０２後、コード・ブロックの変換係数が受け取られ、ビットプレーンに分割される（３０４）。次のステップ３０６で、コード・ブロックのビットプレーンがチェックされ、どのビットプレーンがその中に最上位ノンゼロ・ビットを持っているかが決定される。

【００３２】次いで、本方法はステップ３０８へ進み、このステップで最上位ビットを持つビットプレーンが単一クリーンアップ・パス内でエントロピ符号化される。具体的には、ノンゼロ・ビットの最上位ビットを持つビットプレーンに属するビット・シンボルが各々エントロピ符号化される。これらのビット・シンボルは、例えば図２に図示のように、所定の走査順でエントロピ符号化される。エントロピ符号化ステップ３０８は、入力として符号化対象ビット・シンボルとそのコンテキストとをとる。該コンテキストはＪＰＥＧ２０００に従って好適

に決定される。同時に、シグニフィカンス・プロパゲーション・パスとマグニチュード・リファインメント・パスとの中で処理の対象となる位置のリストが生成され、その結果、本方法は次のビットプレーン内でのビットの効率的処理が可能となる。最初のクリーンアップ・パス３０８中、ビット・シンボルが有意であることが判明した場合、その関連する位置が現在のＭＲリストに追加される。最初のクリーンアップ符号化パス中に本方法は、有意であることが判明した位置の３×３の近傍内の非有意近傍も決定する。次いで、これらの位置は現在のＳＰリストに追加される。

【００３３】ステップ３０８の完了後、本方法は判定ブロック３１０へ進み、そこで処理の対象となるビットプレーンがさらに存在するかどうかのチェックが行われる。もし存在すれば、本方法はステップ３１２へ進み、そこで次のビットプレーンのビット・シンボルが検索される。

【００３４】次いで、本方法は、関連するビット・シンボルを現在のビットプレーンのシグニフィカンス・プロパゲーション・パスで処理するステップ３１４へ進む。このステップ３１４で、現在のＳＰリスト内の位置に対応する現在のビットプレーン内のビット・シンボルがエントロピ符号化される。ＳＰ符号化パス３１４中、ビット・シンボルが有意であることが判明した場合、その関連する位置が新しいＭＲリストに追加される。２回の符号化を避けるために、次のクリーンアップ符号化パス３１８の完了後、この新しいＭＲリストは現在のＭＲリストと併合される。またＳＰ符号化パス３１４中、ビット・シンボルが有意であることが判明したとき、本方法はその有効ビット・シンボルの位置（Ｐ）の非有意近傍を決定する。次いで、本方法は、走査順でその位置（Ｐ）に先行するこれらの非有意近傍の中のいずれの非有意近傍も新しいＳＰリストに追加する。一方、本方法は、走査順でその位置（Ｐ）の後にあるこれらの非有意近傍のいずれの非有意近傍をも現在のＳＰリストに追加する。次いで、本方法は更新された現在のＳＰリストの再索引化を行う。ＳＰ符号化パス３１４中、本方法は、更新された現在のリストＳＰと現在のＭＲリストとを用いてＮリストの生成も開始する。

【００３５】ＳＰ符号化パス３１４の完了後、本方法はマグニチュード・リファインメント符号化パス３１６へ進む。このステップ３１６で、マグニチュード・リファインメント・リスト内の位置に対応する現在のビットプレーン内のすべてのビット・シンボルがエントロピ符号化される。

【００３６】ＭＲ符号化パス３１６の完了後、本方法はクリーンアップ符号化パス３１８へ進む。クリーンアップ符号化ステップ３１８で、（前回の符号化パス中生成された）現在のクリーンアップＮリスト内の位置に対応する現在のビットプレーン内のすべてのビット・シンボルがエントロピ符号化される。クリーンアップ符号化パス３１８中、ビット・シンボルが有意であることが判明したとき、その関連する位置は新しいＭＲリストに追加される。本方法は新しいＭＲリスト内の位置リストの非有意近傍も決定する。次いで、これらの非有意近傍の位置はクリーンアップ・リストから削除され、新しいＳＰリストに追加される。次いで、新しいＭＲリストと新しいＳＰリストはそれぞれ現在のＭＲリストとＳＰリストと併合される。次いで、新しく併合されたＭＲとＳＰリストとは走査順に従って再索引化される。

【００３７】クリーンアップ符号化パス３１８の完了後、本方法は任意のさらなるビットプレーン処理を行うために判定ブロック３１０へ戻る。判定ブロックが真（イエス）を返した場合、すなわち、処理の対象となるさらなるビットプレーンが存在しない場合本方法は終了する（３００）。好適には、本方法は、最上位ビットプレーンから所定の最下位ビットプレーンまで有効ビットを含むビットプレーンを処理することが望ましい。符号化段階中、本方法は、符号化されたビット・ストリームのヘッダへ情報を追加する。該情報によって最上位ビットプレーンと所定の最下位ビットプレーンとが特定される。復号段階では、本方法は符号化されたビット・ストリームの復号化時にこのヘッダ情報を使用する。

【００３８】次に図４を参照すると、図３に図示のエントロピ符号化法で用いられる最初のクリーンアップ符号化ステップ３０８のフローチャートが示されている。クリーンアップ符号化ステップ３０８は、その中に有効ビットを持つ最上位ビットプレーンの処理を開始する。最初のクリーンアップ符号化ステップ３０８の開始時に、シグニフィカンス・プロパゲーション（ＳＰ）リストとマグニチュード・リファインメント（ＭＲ）リストは空であり、すべての係数はこのクリーンアップ・パス中符号化される。このビットプレーンの各係数について、最初のクリーンアップ符号化ステップ３０８は、以下の演算を実行する：（１）係数がこのビットプレーン内で有意である場合、その係数はＭＲリストに追加される。（２）その係数の３×３の近傍のいずれかの係数が有意であって、しかも、その係数自身が有意でない場合、その係数はＳＰリストに追加される。

【００３９】最初のクリーンアップ符号化ステップ３０８の最後に、ＭＲリストとＳＰリストとは完全なものとなり、次のステップ用として使用される準備ができている。最初のクリーンアップ符号化ステップ３０８は、前処理ステップ４１２中このビットプレーン内で係数の各ビット・シンボルを符号化することにより、さらに、現在前処理されている（４１２）係数の有効値に関する情報、および、有意係数の３×３の近傍の範囲内の任意の非有意近傍に関する情報をメモリに書き込むことにより、これらの演算を実行する。以下さらに詳細に説明するように、ビット・シンボルが前処理されている間、該

メモリは更新を受ける継続状態にある。クリーンアップ符号化ステップ３０８は、以下さらに詳細に説明するように、メモリに書き込まれた情報を利用することにより、後処理ステップ４１８中、係数をＭＲリストに加えるべきか、それとも、ＳＰリストに加えるべきかについての決定も行う。

【００４０】最初のクリーンアップ符号化ステップ３０８はサブ・プロシージャであり、有効ビットをその中に持つ最上位ビットプレーンを処理する主要な方法３００によって呼び出される。最初の符号化ステップ３０８はステップ４０２から始まる。次いで、このサブ・プロシージャは変数ｉとｊが双方とも１にセットされるステップ４０４と４０６へ進む。変数ｊは列カウンタとして、変数ｉはストリップ・カウンタとして使用され、最初のクリーンアップ符号化ステップ３０８がＪＰＥＧ２０００の走査順でコード・ブロックのビットプレーンの列とストリップを処理することが可能になる。コード・ブロックはビット・シンボルのＮ列のＭストリップを持つ。ＭとＮの値はコード・ブロックのサイズに応じて変動する。

【００４１】次いで、最初のクリーンアップ符号化ステップ３０８は判定ブロック４０８へ進み、ここで、ストリップ・カウンタｉが１以上か、コード・ブロック内のストリップＭの総数以下であるかどうかのチェックが行われる。判定ブロック４０８がイエス（真）を返した場合、最初のクリーンアップ符号化ステップ３０８は判定ブロック４１０へ進む。次いで、判定ブロック４１０は、列カウンタｊがコード・ブロック内で１以上であるか、列Ｎの総数以下であるかをチェックする。判定ブロック４１０がイエス（真）を返した場合、最初のクリーンアップ符号化ステップ３０８は前処理ステップ４１２へ進み、この前処理ステップ４１２はそのビットプレーンのｉ番目のストリップのｊ番目の列を前処理し、メモリに情報を書き込む。この前処理ステップ４１２について図５を参照しながら以下さらに詳細に後述する。前処理ステップ４１２の完了後、最初のクリーンアップ符号化ステップ３０８はステップ４２０へ進む。判定ブロック４０８と４１０のうちのいずれか一方がノー（偽）を返した場合、最初のクリーンアップ符号化ステップ３０８は前処理ステップ４１２をバイパスし、ステップ４２０へ直接進む。

【００４２】同時に、最初のクリーンアップ符号化ステップ３０８は判定ブロック４１４へ進み、そこでストリップ・カウンタｉが２以上でかつ（Ｍ＋１）以下であるかどうかのチェックが行われる。但し、Ｍはコード・ブロック内のストリップの総数である。判定ブロック４１４がイエス（真）を返した場合、最初のクリーンアップ符号化ステップ３０８は判定ブロック４１６へ進み、列カウンタｊが３以上で、かつ（Ｎ＋２）以下であるかどうかのチェックが行われる。但し、Ｎはストリップ内の列の総数である。判定ブロック４１６がイエス（真）を返した場合、最初のクリーンアップ符号化ステップ３０８は、（ｉ－１）番目のストリップの（ｊ－２）番目の列を後処理する後処理ステップ４１８へ進む。後処理ステップ４１８は、メモリに既に書き込まれた情報を利用して、係数をＭＲリストに加えるべきか、それともＳＰリストに加えるべきかについての決定を行う。この後処理ステップ４１８について図６を参照しながら以下さらに詳細に後述する。後処理ステップ４１８の完了後、最初のクリーンアップ符号化ステップ３０８はステップ４２０へ進む。判定ブロック４１４と４１６のうちいずれか一方がノー（偽）を返した場合、サブ・プロシージャ３０８はこの後処理ステップ４１８をバイパスし、ステップ４２０へ直接進む。このようにして、後処理ステップ４１８が（ｉ－１）番目のストリップの（ｊ－２）番目の列を処理している間、前処理ステップ４１２はｉ番目のストリップのｊ番目の列を処理する。

【００４３】ステップ４２０中、最初のクリーンアップ符号化ステップ３０８が列カウンタｊを１だけ増分する。次いで、最初のクリーンアップ符号化ステップ３０８は判定ブロック４２２へ進み、そこで列カウンタｊが（Ｎ＋２）以下であるかどうかのチェックが行われる。但し、Ｎはストリップ内の列の総数である。判定ブロック４２２がイエス（真）を返した場合、最初のクリーンアップ符号化パス３０８は次の列ｊを処理する判定ブロック４０８と４１４へ戻る。一方、判定ブロック４２２がノー（偽）を返した場合、最初のクリーンアップ符号化ステップ３０８はステップ４２４へ進み、そこでストリップ・カウンタｉは１だけ増分される。次いで、最初のクリーンアップ符号化ステップ３０８が判定ブロック４２６へ進み、そこでストリップ・カウンタｉが（Ｍ＋１）以下であるかどうかのチェックが行われる。但し、Ｍはコード・ブロック内のストリップの総数である。判定ブロック４２６がイエス（真）を返した場合、最初のクリーンアップ符号化ステップ３０８は次のストリップのさらなる処理のためにステップ４０６へ戻る。

【００４４】上述のように、クリーンアップ符号化ステップ３０８はメモリに書き込まれた情報を読み出して、係数をＭＲリストに加えるべきか、それとも、ＳＰリストに加えるべきかについての決定を行う。このメモリは、現在のストリップ・レジスタ、前回のストリップ・レジスタ、次のストリップ・ライン・ストア、前回のストリップ・ライン・ストア、および前回のストリップ・ライン・ストアの形になる。

【００４５】前回のストリップ・レジスタと現在のストリップ・レジスタとは、４つの行とＮ列を有する４×Ｎレジスタ配列として各々構成される。この配列は合計で４Ｎのエントリを有する。前のストリップ・レジスタと現在のストリップ・レジスタとからなる本構成はダブル・バッファリング構成として機能する。前のストリップ

・レジスタの４つの行は、それぞれ、コード・ブロックの（ｉ－１）番目のストリップの連続する４つの行に対応し、前のストリップ・レジスタのＮ列はコード・ブロックの（ｉ－１）番目のストリップのＮ列に対応する。一方、現在のストリップ・レジスタの４つの行はそれぞれ、コード・ブロックのｉ番目のストリップの連続する４つの行に対応し、現在のストリップ・レジスタのＮ列はコード・ブロックのｉ番目のストリップのＮ列に対応する。前のストリップ・レジスタと現在のストリップ・レジスタの各エントリは、コード・ブロックの係数に対応しており、２つのビットＮ、Ｓを有する。Ｓビットは、そのエントリに対応する係数が有意であるかどうかを示す。Ｎビットはそのエントリに対応する係数が有意係数の３×３の近傍に存在するかどうかを示す。現在のストリップ・レジスタは、前処理ステップ４１２中継続的に更新される。例えば、ｉ番目のストリップのｊ番目の列の前処理４１２中、現在のストリップ・レジスタのｊ番目の列のＳビットが更新され、コード・ブロックのｉ番目のストリップのｊ番目の列の係数の有効値に応じて対応するＮビットが列（ｊ－１）、ｊ、（ｊ＋１）について更新される。ｉ番目のストリップの最後の列の前処理（ステップ４２４中）の完了後、現在のストリップ・レジスタの内容が前のストリップ・レジスタへ出力され、現在のストリップ・レジスタのＮ、Ｓビットはゼロにリセットされる。したがって、（ｉ＋１）番目のストリップの前処理中の前のストリップ・レジスタには、ｉ番目のストリップに対する現在のストリップ・レジスタの完全な更新が含まれる。前処理ステップ４１２がｉ番目のストリップ用の現在のストリップ・レジスタを更新中同時に、後処理４１８は前のストリップ・レジスタからＮ、Ｓビットを読み出している。

【００４６】前回のストリップ・ライン・ストアと次のストリップ・ライン・ストアとは、１×Ｎライン・ストアとして各々構成される。前回のストリップ・ライン・ストアと次のストリップ・ライン・ストアとから成る本構成はダブル・バッファリング構成として機能する。前回のストリップ・ライン・ストアはｉ番目のストリップの第１の行に対応する。前回のストリップ・ライン・ストアは、その位置の３×３の近傍内の（ｉ－１）番目のストリップの最後の行に有意係数を持つ位置であって、ｉ番目のストリップの第１の行内の非有意係数の位置の列番号を格納する。次のストリップ・ライン・ストアの行は、（ｉ＋１）番目のストリップの第１の行に対応する。次のストリップ・ライン・ストアは、その位置の３×３の近傍内のｉ番目のストリップの最後の行に有意係数を持つ位置であって、（ｉ＋１）番目のストリップの第１の行の非有意係数の位置の列番号を格納する。いずれの場合も、列番号はＪＰＥＧ２０００の走査順でライン・ストアの先頭から次々にライン・ストア内に格納される。いずれの１回についても、次のストリップ・ライン・ストアと前回のストリップ・ライン・ストアの双方ともＮ列番号まで格納することができる。Ｎ未満の列番号しかない場合、ライン・ストアの残りのエントリと最後の部分はブランク状態のままに放置される。次のストリップ・ライン・ストアは、前処理ステップ４１２中継続的に更新される。例えば、ｉ番目のストリップのｊ番目の列の前処理４１２中、ｉ番目のストリップのｊ番目の列の有意係数の３×３の近傍の範囲内に在る、（ｉ＋１）番目のストリップの第１の行の範囲内のいずれの位置の列番号も、次のストリップ・ライン・ストアに追加される。ｉ番目のストリップの最後の列（ステップ４２４中）の前処理の完了後、次のストリップ・ライン・ストアの内容は前回のストリップ・ライン・ストアへ出力され、次のストリップ・ライン・ストアのエントリはゼロにリセットされる。したがって、（ｉ＋１）番目のストリップの前処理中の前回のストリップ・ライン・ストアには、ｉ番目のストリップに対する次のストリップ・ライン・ストアの完全な更新が含まれる。ｉ番目のストリップの処理中の前処理ステップ４１２は、（ｉ＋１）番目用のストリップの次のストリップ・ライン・ストアを更新し、ｉ番目のストリップの前回のストリップ・ライン・ストアを読み出す。

【００４７】前回のストリップ・ライン・ストアは１×４ライン・ストアとして構成される。前回のストリップ・ライン・ストアの行は、列（ｊ－２）、（ｊ－１）、ｊ、（ｊ＋１）の（ｉ－１）番目のストリップの最後の行に対応する。前回のストリップ・ライン・ストアは、その位置の３×３の近傍のｉ番目のストリップの第１の行に有意係数を持つ位置であって、（ｉ－１）番目のストリップの最後の行の非有意係数の位置の列番号を格納する。列番号はＪＰＥＧ２０００の走査順で格納される。前回のストリップ・ライン・ストアは前処理ステップ４１２中継続的に更新される。例えば、ｉ番目のストリップのｊ番目の列の前処理４１２中、ｉ番目のストリップのｊ番目の列の有意係数の３×３の近傍の範囲内にある（ｉ－１）番目のストリップの最後の行の範囲内の任意の位置の列番号は、（予めそこに存在しない場合）前回のストリップ・ライン・ストアに追加される。同時に、後処理ステップ４１８は、前回のストリップ・ライン・ストアの第１のエントリを読み出している。これらの列番号は、ライン・ストアの先頭から始まるライン・ストアの前回のストリップ・ライン・ストアの中にＪＰＥＧ２０００の走査順に次々に格納される。いずれの１回についても、前回のストリップ・ライン・ストアは４までの列番号を格納することができる。４未満の列番号しかない場合、前回のストリップ・ライン・ストアの残りのエントリと最後の部分はブランク状態のままに放置される。

【００４８】次に図５を参照して、図４に描かれているようなｉ番目のストリップのｊ番目の列の前処理ステッ

プ４１２のフローチャートが更に詳細に説明する。前処理ステップ４１２はステップ５０２から始まり、ステップ５０４へ進む。ここで、ｉ番目のストリップのｊ番目の列の第１のビット・シンボルが検索される。この検索対象ビット・シンボルは、ｊ番目の列でかつｉ番目のストリップのＪＰＥＧ２０００の走査順で最初（即ち、行１）のビット・シンボルである。次いで、前処理ステップ４１２はステップ５０６へ進み、そこでＪＰＥＧ２０００に従って検索対象ビット・シンボルに対する符号化とコンテキスト生成とが行われる。符号化の完了後、ステップ５０６は必要に応じて有効値テーブルの更新も行う。係数の第１のノンゼロ・ビットプレーン値が符号化されると、係数の有意状態は（１）に変化し、テーブルの配列の対応するエントリは（１）にリセットされる。具体的には、ある係数に属する現在の符号化されたビット・シンボルが（１）である場合、有効値テーブルに格納される係数の有効値は（１）にセットされる。ステップ５０６の完了後、前処理ステップ４１２は判定ブロック５０８へ進み、そこで符号化されたビットが有意な係数に属するかどうかのチェックが行われる。判定ブロック５０８がイエス（真）を返した場合、前処理ステップ４１２はステップ５１０へ進み、そこで検索対象のビット・シンボルに対応するｊ番目の列であって、現在のストリップ・レジスタのｊ番目の列のビットＳは（１）にセットされる。判定ブロック５０８がノー（偽）を返した場合（即ち、係数が有意ではない場合）、前処理ステップ４１２は判定ブロック５３４へ直接進む。

【００４９】ステップ５１０の後、前処理ステップ４１２はループ５１４から５２６へ進み、そこで現在の有意係数の３×３の近傍内の各非有意近傍が順に処理される。ステップ５１４中、前処理ステップ４１２は、（もし存在する場合には）現在の有意係数の非有意近傍を取得する。ステップ５１４は、有効値テーブルを調べることにより現在の有意係数の非有意近傍を決定する。非有意近傍が存在しなければ、前処理ステップ４１２は判定ブロック５３４へ直接進む（図示せず）。一方、非有意近傍が存在する場合、前処理ステップ４１２は判定ブロック５１６へ進み、そこで現在の近傍が前回のストリップ（ｉ－１）内にあるかどうかのチェックが行われる。判定ブロック５１６がイエス（真）を返した場合、前処理ステップ４１２はステップ５１８へ進み、そこで現在の近傍の列番号が前回のストリップ・ライン・ストアに追加される。ステップ５１８の後、前処理ステップ４１２は判定ブロック５２６へ進む。一方、判定ブロック５１６がノー（偽）を返した場合、前処理ステップ４１２は判定ブロック５２０へ進み、そこで現在の近傍が次のストリップ（ｉ＋１）にあるかどうかのチェックが行われる。判定ブロック５２０がイエス（真）を返した場合、前処理ステップ４１２はステップ５２２へ進み、そこで列番号が次のストリップ・ライン・ストアに追加される。ステップ５２２後、前処理ステップ４１２は判定ブロック５２６へ進む。一方、判定ブロック５２０がノー（偽）を返した場合、前処理ステップ４１２はステップ５２４へ進む。ステップ５２４中、前処理ステップ４１２は、現在の近傍に対応する現在のストリップ・レジスタのエントリ内でビットＮを（１）にセットする。ステップ５２４後、前処理ステップ４１２は判定ブロック５２６へ進む。判定ブロック５２６は、ループ５１４～５２６によってまだ処理されていない近傍が、現在の有意係数の３×３の近傍内にさらに存在するか否かをチェックする。判定ブロック５２６がイエス（真）を返した場合、現在の有意係数の次の近傍の処理を行うために前処理ステップ４１２はステップ５１４へ戻る。判定ブロック５２６がノー（偽）を返した場合、前処理ステップはステップ５２８へ進む。

【００５０】ステップ５２８中、前処理ステップ４１２は、ＪＰＥＧ２０００の走査順に従う前回のストリップ・ライン・ストアの第１のエントリである前回のストリップ・ライン・ストアの先頭を検索し、それを変数headSPに割り振る。ｉ番目のストリップの最初の前処理ステップ４１２中（即ち、ｊ＝１）、前回のストリップ・ライン・ストアは、ｉ番目のストリップの第１の行に対応し、次いで、それらの３×３の近傍の（ｉ－１）番目のストリップ内の有意係数を持つ位置の列番号を格納する。従って、ｉ番目のストリップの最初の前処理ステップ４１２（即ち、ｊ＝１）中、ある位置のｉ番目のストリップの第１の行の前回のストリップ・ライン・ストア内の変数headSPにＪＰＥＧ２０００の走査順で格納された第１列番号が含まれる。この位置は（ｉ－１）番目のストリップ内のその位置の３×３の近傍に有意係数を持つ。ステップ５２８後、前処理ステップ４１２は判定ブロック５３０へ進み、そこでｊ＝headSPかどうかのチェックが行われる。判定ブロック５３０がイエス（真）を返した場合、前処理ステップ４１２はステップ５３２へ進み、そこでＮビットは位置（１，ｊ）に対応する現在のストリップ・レジスタのエントリ用として（１）にセットされる。このようにして、（ｉ－１）番目のストリップ内の有意係数のｉ番目のストリップの非有意近傍がそれらの対応するＮビットをセットすることにより現在のストリップ・レジスタ内に特定される。また、ステップ５３２中、前処理ステップ４１２によって前回のストリップ・ライン・ストアの現在の先頭が取り除かれる。さらに、残りの列番号はすべて前回のストリップ・ライン・ストアの先頭へ一度にシフトされる。ステップ５３２の完了後、前処理ステップ４１２は判定ブロック５３４へ進む。一方、判定ブロック５３０がノー（偽）を返した場合、前処理ステップ４１２は判定ブロック５３４へ直接進む。当業者には明らかなように、前回のストリップ・ライン・ストアはＮまでの列番号を最初に格納する。前処理ステップ４１２が各列ｊ＝１，…，Ｎについ

て呼び出されると、前処理ステップ４１２は、判定ブロック５３０の間、前回のストリップ・ライン・ストア内に対応する列番号が存在するかどうかをチェックする。もし対応する列番号が存在すれば、非有意近傍はその列の第１の行に存在するものとして特定され、現在のストリップ・レジスタに入力され、前回のストリップ・ライン・ストアから取り除かれる。

【００５１】判定ブロック５３４は、前処理ステップ４１２によってまだ処理されていないビット・シンボルがｊ番目の列の中に更に存在するかどうかを判定する。判定ブロック５３４がイエス（真）を返した場合、前処理ステップ４１２はステップ５３６へ進み、そこで列内の次のビット・シンボルが検索される。次いで、前処理ステップ４１２は、この検索されたビット・シンボルを処理するステップ５０６へ戻る。一方、列ｊの中にそれ以上処理すべきビット・シンボルが存在しなければ、判定ブロック５３４はノー（偽）を返し、前処理ステップ４１２は終了し（５３８）、さらなる処理を行うために最初のクリーンアップ符号化ステップ３０８へ戻る。

【００５２】次に図６に目を転じると、図４に描かれているような、（ｉ－１）番目のストリップの（ｊ－２）番目の列の後処理ステップ４１８のフローチャートがさらに詳細に示されている。前述したように、前処理ステップ４１２がｉ番目のストリップのｊ番目の列を処理している間、後処理ステップ４１８は（ｉ－１）番目のストリップの（ｊ－２）番目の列を処理する。

【００５３】後処理ステップ４１８はステップ６０２から始まり、ステップ６０４へ進み、そこで変数position1は（1，ｊ－２）にセットされる。この変数position1は、後処理ステップによって処理される前のストリップ・レジスタ内に現在位置の（行、列）座標を格納する。ステップ６０４の完了後、後処理ステップ４１８はステップ６１０へ進み、そこで前回のストリップ・ライン・ストアの先頭に格納されたエントリが検索され、変数headSPの中に格納される。この変数headSPは、或る位置の３×３の近傍に在るｉ番目のストリップの中に有意係数を持つ（ｉ－１）番目のストリップの中に位置の列番号を格納する。前回のストリップ・ライン・ストアが空の場合、ステップ６０６は変数headSPをヌルにセットする。

【００５４】ステップ６０６の完了後、後処理ステップ４１８は判定ブロック６０８へ進み、そこで変数headSP＝（ｊ－２）かどうかのチェックが行われる。判定ブロック６０８がイエス（真）を返した場合、後処理ステップ４１８はステップ６１０へ進み、そこで変数positon2は（4、headSP）にセットされる。この変数positon2は、（ｉ－１）番目のストリップ内のビット・シンボルの（行、列）を示し、該シンボルはｉ番目のストリップ内のその３×３の近傍に有意係数を持っている。判定ブロック６０８がノー（偽）を返した場合、この変数positon2はヌルにセットされる（６１２）。ステップ６１２の完了後、後処理ステップ４１８はステップ６１４へ進む。

【００５５】ステップ６１４中、後処理ステップ４１８は変数position1とpositon2を比較し、ＪＰＥＧ２０００の走査順でposition1がpositon2より前に存在するかどうかを判定する。存在するならば、後処理ステップ４１８は変数winning_position内に変数position1を格納する。存在しなければ、後処理ステップ４１８は変数winning_positionに変数positon2を格納する。ステップ６１４の完了後、後処理ステップ４１８は判定ブロック６１６へ進み、そこで変数winning_positionに格納された座標が変数positon2（即ち、前回のストリップ・ライン・ストア）であるかどうかのチェックが行われる。

【００５６】判定ブロック６１６がイエス（真）を返した場合、後処理ステップはステップ６１８へ進み、そこでheadSPに対応する前回のストリップ・ライン・ストアの先頭が取り除かれる。また、前回のストリップ・ライン・ストアの残りのエントリは先頭へ１だけ各々シフトされる。ステップ６１８の完了後、後処理ステップはステップ６２０へ進み、そこで変数winning_positionに格納された位置に対応する前のストリップ・レジスタ内の位置にあるＮビットは（1）にセットされる。ステップ６２０の完了後、後処理ステップ４１８は判定ブロック６２２へ進む。判定ブロック６１２がノー（偽）を返した場合、後処理ステップは判定ブロック６２２へ直接進む。

【００５７】判定ブロック６２２は、変数winning_positionに格納された位置に対応する前のストリップ・レジスタ内の位置のＳビットが（1）であるかどうかをチェックする。判定ブロック６２２がイエス（真）を返した場合、後処理ステップ４１８はステップ６２４へ進み、そこで変数winning_positionに格納された位置はＭＲリストに追加される。ステップ６２４の完了後、後処理ステップ４１８は判定ブロック６３０へ進む。判定ブロック６２２がノー（偽）を返した場合、後処理ステップ４１８は判定ブロック６２６へ進む。判定ブロック６２６は、変数winning_positionに格納された位置に対応する前のストリップ・レジスタ内の位置のＮビットが（1）であるかどうかをチェックする。判定ブロック６２６がイエス（真）を返した場合、後処理ステップ４１８はステップ６２８へ進み、そこで変数winning_positionに格納された位置はＳＰリストに追加される。ステップ６２８の完了後、後処理ステップ４１８は判定ブロック６３０へ進む。判定ブロック６２６がノー（偽）を返した場合、後処理ステップ４１８は判定ブロック６３０へ直接進む。

【００５８】判定ブロック６３０は、（ｊ－２）番目の列の中に処理の対象となる位置がさらに存在するかどうかをチェックする。すなわち、判定ブロック６３０は、

変数winning_positionに格納された現在位置の行が４に等しいかどうかをチェックする。判定ブロック６３０がノー（偽）を返した場合、後処理ステップ４１８はステップ６３２へ進み、そこで変数position1内の行番号は（１）だけ増分される。ステップ６３２後、後処理ステップ４１８はステップ６０６へ進み、そこで変数position1に格納された新しい値に対して後処理４１８が再開される。一方、判定ブロック６３０がイエス（真）を返した場合、後処理ステップ４１８は終了し（６３４）、更なる処理を行うためにクリーンアップ符号化ステップ３０８へ戻る。

【００５９】上記から解るように、ｉ番目のストリップに対する前処理ステップ４１２は、現在のストリップ・レジスタ内に格納するために現在のＮ、Ｓビットを生成する。これらのＮ、Ｓビットは、ｉ番目のストリップ用として前処理ステップの終了時に存在し、次いで、前のストリップ・レジスタへ出力される。ｉ番目のストリップの前処理ステップ４１２と同時に、後処理ステップ４１８は、前のストリップ・レジスタに現在格納されている（ｉ－１）番目のストリップのＮ、Ｓビットを処理する。このＮ、Ｓビットは、前回の前処理ステップ４１２中（ｉ－１）番目のストリップ用として生成されたものである。ｉ番目のストリップの前処理ステップ４１２は、ｉ番目のストリップ内の有意係数とその非有意近傍とを特定し、現在のストリップ・レジスタの対応する位置のＮ、Ｓビットとしてそれらを格納する。前処理ステップ４１２は、（ｉ－１）番目のストリップ内の有意係数のｉ番目のストリップの第１の行のいずれの非有意近傍も特定し、現在のレジスタの対応する位置にＮビットとしてこれらを格納する。ｉ番目のストリップの前処理ステップ４１２は、前回のストリップ・ライン・ストアに現在格納されている任意のこのような非有意近傍の列番号を検索することにより後者の処理を行う。これらの非有意近傍は（ｉ－１）番目のストリップの前回の前処理ステップで特定され、次のストリップ・ライン・ストアに格納され、（ｉ－１）番目のストリップの前処理ステップの最後に、前回のストリップ・ライン・ストアへ出力される。ｉ番目のストリップの前処理ステップ４１２によって、ｉ番目のストリップの第１の行内の有意係数に起因する非有意近傍であって、（ｉ－１）番目のストリップの最後の行内のいずれの非有意近傍も特定される。前処理ステップ４１２は、これらの非有意近傍の列番号を前回のストリップ・ライン・ストアに追加する。次いで、（ｉ－１）番目のストリップに対する後処理ステップ４１８は、前回のストリップ・ライン・ストアの中からこれらの列番号を検索し、前のストリップ・レジスタ内の任意の対応するＮビットを更新する。（ｉ－１）番目のストリップに対する後処理ステップ４１８は、更新された前のストリップ・レジスタに格納されているＮ、Ｓビットを利用して、ＳＰリスト又はＭＲリストに追加が可能な位置を特定する。具体的には、１にセットされたＳビットを有する更新された前のストリップ・レジスタ内の位置はＭＲリストの中に格納され、Ｓ＝１とＮ＝０を持つ更新された前のストリップ・レジスタ内の位置がＳＰリストの中に格納される。

【００６０】図３に戻ると、最初のクリーンアップ符号化ステップ３０８の完了後、本方法は判定ブロック３１０へ進み、そこで処理の対象となるビットプレーンがさらに存在するかどうかのチェックが行われる。もし存在すれば、本方法はステップ３１２へ進み、そこで次のビットプレーンのビット・シンボルが検索される。次いで、本方法は、現在のビットプレーンのシグニフィカンス・プロパゲーション・パスで、関連するビット・シンボルの処理を行うステップ３１４へ進む。

【００６１】次に図７Ａと図７Ｂに目を転じると、図３のシグニフィカンス・プロパゲーション（ＳＰ）符号化ステップ３１４のフローチャートがさらに詳細に示されている。この符号化ステップ３１４は、最初のクリーンアップ符号化ステップ３０８の後に最初続き、その後、初期ビットプレーンの後の各ビットプレーン用のクリーンアップ符号化ステップ３１８に続く。ＳＰ符号化ステップ３１４は、エントロピ符号化法３００と呼ばれサブ・プロシージャの形をとる。この符号化ステップ３１４は、現在のＳＰリスト（本明細書では、以後オリジナルのＳＰリストと呼ぶ）が処理を行うためにＳＰ符号化ステップ３１４へ渡されたところから始まる（７０２）。前回の最初のクリーンアップ・ステップ３０８又はクリーンアップ符号化ステップ３１８によって、このオリジナルのＳＰリストは生成される。ＳＰ符号化ステップ３１４は、ＳＰ符号化ステップ３１４へ渡されるオリジナルのＳＰリストと、近傍ＳＰリストと、次のストリップＳＰリストと、前回のストリップＳＰリストと、新しいＳＰリストの５つのリストを利用する。ＳＰ符号化パス３１４によって後者の４つのリストが生成される。近傍ＳＰリストと、前回のストリップＳＰリストと、次のストリップＳＰリストとは、それらのリストの利用がＳＰ符号化ステップ３１４に限定されるという点でローカルなものである。一方、生成された新しいＳＰリストと更新されたオリジナルのＳＰリストとはステップ３１８中さらなる処理を行うためにエントロピ符号化法へ渡される。

【００６２】現在のＳＰ符号化ステップ２１４中後で有意になる係数である、このリストに格納されている位置に対応する任意の係数をタグ付けによって特定するために、ＳＰ符号化ステップ３１４中オリジナルのＳＰリストの更新を行うことができる。この目的のために、オリジナルのＳＰリストの各エントリは、そのロケーション３×３の近傍に有意係数を有するロケーションであって、非有意係数の現在のビットプレーン内のビット・シンボルのロケーションを示す（行、列）座標と、現在の

ＳＰ符号化ステップ２１４後にビット・シンボルの有効値を示すフラグとを有する。

【００６３】新しいＳＰリストは、そのロケーションの３×３の近傍に任意の新しい有意係数を持つ非有意係数のビット・シンボルのロケーションの（行、列）座標を有する。これらの新しい有意係数は、現在のビットプレーン内にビット・シンボルを持つ、符号化された有意係数であり、これらの有意係数の有意状態は現在のＳＰ符号化ステップ３１４中、1に変化する（有意へ変わる）。現在のＳＰ符号化ステップ３１４中後で有意になる係数である、このリストに格納された位置に対応する任意の係数をタグ付けによって特定するために、ＳＰ符号化ステップ３１４中新しいＳＰリストの更新を行うことができる。この目的のために、新しいＳＰリストの各エントリは、そのロケーションの３×３の近傍に新しい有意係数を有するロケーションであって、非有意係数のビット・シンボルのロケーションの（行、列）座標と、現在のＳＰ符号化ステップ３１４後にビット・シンボルの有効値を示すフラグとを有する。

【００６４】近傍ＳＰリストはいくつかのビット・シンボルのコード・ブロックの範囲内にロケーションの（行、列）座標を有する。近傍ＳＰリストは新しいＳＰリストのサブセットである。すなわち、近傍ＳＰリスト内のすべての位置はこの新しいリストの中に含まれる。近傍ＳＰリストには、ＳＰ符号化ステップ３１４中復号化されたばかりの現在の有意係数の３×３の近傍の範囲内に、非有意係数の位置（但し、必ずしも全ての位置ではない）が含まれる。近傍ＳＰリストは、現在の有意係数と同じコード・ブロックのストリップ内に在り、かつ、ＪＰＥＧ２０００の走査順で現在の有意係数の後に在る、３×３の近傍の範囲内の位置しか格納しない。

【００６５】次のストリップＳＰリストは、ＳＰ符号化ステップ３１４中有意であることが判明した、現在のストリップ内の係数の３×３の近傍の範囲内の次のストリップの第１の行の範囲内に非有意係数のロケーションの列座標を有する。次のストリップＳＰリストは、近傍ＳＰリスト内のすべての位置が新しいリストに含まれるという点で、新しいＳＰリストのサブセットである。

【００６６】前回のストリップＳＰリストは、ＳＰ符号化ステップ３１４中有意であることが判明した、前回のストリップ内の係数の３×３の近傍の範囲内の現在のストリップの第１の行の範囲内に非有意係数のロケーションの列座標を有する。

【００６７】開始後、ＳＰ符号化ステップ３１４は初期化ステップ７０４へ進み、そこで近傍ＳＰリストに対するエントリと、次のストリップＳＰリストと、前回のストリップＳＰリストと、新しいＳＰリストとがゼロ（０）にリセットされる。初期化ステップ７０４後、サブ・プロシージャは判定ブロック７０６へ進み、そこでオリジナルのＳＰリスト内に何らかの位置が存在するかどうかのチェックが行われる。判定ブロック７０６がノー（偽）を返した場合、ＳＰ符号化ステップ３１４は終了し（７１４）、エントロピ符号化法はＭＲ符号化ステップ３１６へ進む。コード・ブロックの範囲内のすべての位置が有意で、ＭＲ符号化ステップ３１６中符号化の対象となる場合、上記の事態が生じることもある。判定ブロック７０６がイエス（真）を返した場合、ＳＰ符号化ステップ３１４はループ（ステップ７１０～７２８）と、クリーンアップ・リスト生成ステップ７０８とへ同時に進む。ＳＰ符号化ステップ３１４がＳＰ位置を符号化し、上述のＳＰリストを生成している間、クリーンアップ・リスト生成ステップ７０８はクリーンアップ・リストを生成する。以下より詳細にクリーンアップ・リスト生成ステップ７０８について説明する。

【００６８】ループ（７１０～７２８）は、ループの各パスについてオリジナルのＳＰリスト内と、近傍ＳＰリスト内と、前回のストリップＳＰリスト内の各位置を処理する。ループの任意のパス中にＳＰ符号化ステップ３１４は、オリジナルのＳＰリスト内と、近傍ＳＰリスト内と、前回のストリップＳＰリスト内の次の位置であって、前回のパス中検索されなかった次の位置を検索する（７２０）。これらのリスト内の位置は、ＪＰＥＧ２０００の走査順に従ってループのパス中次々に検索される。これらの位置の検索方法について図８を参照しながら以下さらに詳細に後述する。一方、これらのリストの範囲内に前に検索されなかった位置がそれ以上存在しなかった場合、ＳＰ符号化ステップ３１４はループ（ステップ７１０～７２８）から抜けてステップ７１２へ進む。

【００６９】初期化ステップ７０４の完了後、ＳＰ符号化ステップ３１４は判定ブロック７０６へ進み、そこでオリジナルのＳＰリスト内、近傍ＳＰリスト内及び前回のストリップＳＰリスト内に位置がさらに存在するかどうかのチェックが行われる。これらのリストの範囲内に残りの位置が存在すれば、判定ブロックはイエス（真）を返し、ＳＰ符号化ステップ３１４は判定ブロック７１０へ進む。

【００７０】判定ブロック７１６はコード・ブロックの現在のストリップ内、オリジナルのＳＰリスト内、近傍ＳＰリスト内及び前回のストリップＳＰリスト内に、前に検索されなかった位置がさらに存在するかどうかをチェックする。ＳＰ符号化ステップ３１４は、第１のストリップからコード・ブロックのストリップを処理し、ＪＰＥＧ２０００に記載されている走査順で、最後のストリップまでずっと処理を継続する。判定ブロック７１６の最初のパス中、現在のストリップはコード・ブロックの第１のストリップである。判定ブロック７１６がノー（偽）を返した場合、ＳＰ符号化ステップ３１４はステップ７１８へ進み、そこで次のストリップＳＰリスト内の列位置が前回のストリップＳＰリストへ出力され、次

のストリップＳＰリストのエントリはゼロにセットされる。前述したように、次のストリップＳＰリストは、ＳＰ符号化ステップ３１４中有意であることが判明した係数である、現在のストリップ内の係数の３×３の近傍の範囲内の次のストリップの第１の行の範囲内の非有意係数のロケーションの列座標を有する。前回のストリップＳＰリストは、ＳＰ符号化ステップ３１４中有意であることが判明した係数である、前回のストリップ内の係数の３×３の近傍の範囲内の現在のストリップの第１の行の範囲内の非有意係数のロケーションの列座標を有する。ＳＰ符号化ステップ３１４は、次のストリップＳＰリストに列番号を追加し（ステップ７３８）、ＳＰ符号化コード・ブロックの任意のストリップの処理中、前回のストリップＳＰリストの中から列番号を検索する。現在のストリップの範囲内にそれ以上位置が存在しなければ、次のストリップＳＰリストの列番号が前回のストリップＳＰリストへ出力され（７１８）、次のストリップＳＰリストのエントリがゼロにセットされ、次のストリップが現在のストリップになる。ステップ７１８の完了後、ＳＰ符号化ステップ３１４はステップ７２０へ進む。

【００７１】一方、判定ブロック７１６がイエス（真）を返した場合、ＳＰ符号化ステップ３１４はステップ７２０へ直接進む。ステップ７２０中、ＳＰ符号化ステップ３１４はオリジナルのＳＰリスト内と、近傍ＳＰリスト内と、前回のストリップＳＰリスト内の次の位置であって、ループの前回のパス中検索されなかった位置を検索する。以下これについて図８を参照しながらより詳細に説明する。ステップ７２０の完了後、ＳＰ符号化ステップ３１４はステップ７２２へ進み、そこでコンテキスト生成とビット・シンボルの符号化とがその（検索）位置において行われる。この符号化とコンテキスト生成とは、ＪＰＥＧ２０００に従って好適に行われる。符号化の完了後、ステップ７２２は必要な場合に有効値テーブルの更新も行う。係数の第１のノンゼロ・ビットプレーン値が符号化され、テーブルの配列の対応するエントリが（１）にリセットされた場合、係数の有意状態は（１）に変化する。具体的には、ある係数に属する現在の符号化されたビット・シンボルが（１）である場合、有効値テーブルに格納されるその係数の有効値は（１）にセットされる。

【００７２】符号化ステップ７２２の完了後、ＳＰ符号化ステップ３１４は判定ブロック７２４へ進む。判定ブロック７２４では、その（検索）位置における符号化されたビット・シンボルが（１）の有意状態を持っているかどうかのチェックが行われる。判定ブロック７２４がノー（偽）を返した場合、ＳＰ符号化ステップ３１４はステップ７１０へ戻り、そこで、オリジナルのＳＰリスト内と、近傍ＳＰリスト内と、前回のストリップＳＰリスト内の次の位置であって、ループの前回のパス中検索されなかった次の位置が検索される。一方、判定ブロック７２４がイエス（真）を返した場合、ＳＰ符号化ステップ３１４はステップ７２６へ進み、そこでオリジナルのリスト内又は新しいＳＰリスト内の対応する（検索された）位置のフラグに有効なものとしてタグが付けられる。このようにして、次のクリーンアップ符号化ステップ３１８中、ＭＲリストへの後続する追加を行うためにこの位置が特定される。

【００７３】ステップ７２６の完了後、ＳＰ符号化ステップ３１４は、新しいＳＰに非有意近傍を追加するための処理へこの非有意近傍の（行、列）座標を渡す（７３２）。これについては図１３Ａと図１３Ｂを参照しながらさらに詳細に後述する。この処理はＳＰ符号化ステップ３１４の開始から始まり、ＳＰ符号化ステップ３１４の完了で終了する。

【００７４】ステップ７２６の完了後、ＳＰ符号化ステップ３１４はサブループ７２８～７３８、７４０へ進み、そこで現在の符号化されたビット・シンボルを有する現在の有意係数の３×３の近傍内の非有意係数がサブループの各パスについて順に処理される。サブループ７２８～７３８、７４０は判定ブロック７２８を有し、この判定ブロックで、サブループによって前に処理されなかったような、現在の有意係数の非有意近傍がさらに存在するかどうかのチェックが行われる。判定ブロック７２８は有効値テーブルを参照してこの判定を行う。判定ブロック７２８がノー（偽）を返した場合、ＳＰ符号化ステップ３１４はさらなる処理を行うために判定ブロック７１０へ戻る。一方、判定ブロック７２８がイエス（真）を返した場合、ＳＰ符号化ステップ３１４はステップ７３０へ進み、そこで（行、列）座標としての非有意近傍の位置が有効値テーブルから検索される。ステップ７３０の完了後、ＳＰ符号化ステップ３１４は判定ブロック７３４へ進み、そこで現在の非有意近傍の位置が、ＪＰＥＧ２０００の走査順で現在の符号化されたビット・シンボルを有する係数の位置の後に在るかどうかのチェックが行われる。判定ブロック７３４がノー（偽）を返した場合、さらなる処理を行うために、ＳＰ符号化ステップ３１４は判定ブロック７２８へ戻る。一方、判定ブロック７３４がイエス（真）を返した場合、ＳＰ符号化ステップ３１４は判定ブロック７３６へ進み、そこで現在の符号化されたビット・シンボルが属する相手先の現在のストリップ後、現在の非有意近傍が次のストリップに属するかどうかのチェックが行われる。判定ブロック７３６がイエス（真）を返した場合、ＳＰ符号化ステップ３１４はステップ７３８へ進み、そこで現在の非有意近傍の位置の列番号が次のストリップＳＰリストの最後に追加される。ステップ７３８の完了後、ＳＰ符号化ステップ３１４はさらなる処理を行うために判定ブロック７２８へ戻る。一方、判定ブロック７３８がノー（偽）を返した場合、ＳＰ符号化ステップ３１４

はステップ７４０へ進み、そこで現在の非有意近傍の位置の（行、列）座標が近傍ＳＰリストの最後に追加される。ステップ７４０の完了後、ＳＰ符号化ステップ３１４はさらなる処理を行うために判定ブロック７２８へ戻る。

【００７５】次に図８に目を転じると、ＳＰ符号化ステップ３１４のステップ７２０のフローチャートがさらに詳細に示されている。ステップ７２０はＪＰＥＧ２０００の走査順に従って、オリジナルのＳＰリスト内、近傍ＳＰリスト内、または、前回のストリップＳＰリスト内に格納される第１の位置であって、前に検索され、処理されなかった第１の位置を検索する。以上から明らかなように、オリジナルのＳＰリスト内、近傍ＳＰリスト内、及び、前回のストリップＳＰリスト内に格納された位置は、ＪＰＥＧ２０００に従ってソートされる。ステップ７２０に対する第１のコール中、３つのヘッド・ポインタはそれぞれ、オリジナルのＳＰリストへの第１のエントリと、近傍ＳＰリストへの第１のエントリと、前回のストリップＳＰリストへの第１のエントリを指している。各後続するコールについて、これらのヘッド・ポインタの中の１つはリスト内の次の項目へ移動する。ステップ７１８が次のストリップＳＰリストの内容を前回のストリップＳＰリストへ出力した場合、前回のストリップＳＰリストのためのヘッド・ポインタは、前回のストリップＳＰリストの第１のエントリにリセットされる。ヘッド・ポインタの位置は、現在のエントロピ・ステップ３１４内の、ステップ７２０への各コールの間で保持される。現在のエントロピ・ステップ３１４が完了されるとすぐにヘッド・ポインタはリセットされる。

【００７６】ステップ７２０はステップ８０２から始まり、判定ブロック８０４、８０６、８０８へ同時に進む。判定ブロック８０４で、ステップ７２０は、ヘッド・ポインタがオリジナルのＳＰリスト内の有意な（即ち、ヌルでない）位置を指しているかどうかをチェックする。判定ブロック８０４がイエス（真）を返した場合、ステップ７２０はステップ８１２へ進み、そこでステップ７２０は、ヘッド・ポインタが指しているオリジナルのＳＰリスト内のその位置の（行、列）座標を検索し、変数origSPにこの座標を格納する。一方、判定ブロック８０４がノー（偽）を返した場合、ステップ７２０は変数origSPをヌルにセットする。判定ブロック８０６で、ステップ７２０はヘッド・ポインタが近傍ＳＰリスト内で有意な（即ち、ヌルでない）位置を指しているかどうかをチェックする。判定ブロック８０６がイエス（真）を返した場合、ステップ７２０はステップ８１６へ進み、そこでステップ７２０は、ヘッド・ポインタが指している、近傍ＳＰリスト内のその位置の（行、列）座標を検索し、変数neighSPにこの座標を格納する。一方、判定ブロック８０６がノー（偽）を返した場合、ステップ７２０は変数neighSPをヌルにセットする。判定ブロック８０８で、ステップ７２０はヘッド・ポインタが前回のストリップＳＰリストの中で有意な（即ち、ヌルでない）位置を指しているかどうかをチェックする。判定ブロック８０８がイエス（真）を返した場合、ステップ７２０はステップ８２０へ進み、そこでステップ７２０はヘッド・ポインタが指している前回のストリップＳＰリスト内のその位置の列番号を検索する。この列番号は変数nextstripSP内の座標（１、列番号）として格納される。上述のように、前回のＳＰリストは、前回のストリップのＳＰ符号化ステップ３１４中符号化された有意係数に起因する非有意近傍であって、ストリップの第１の行の任意の非有意近傍の列番号を格納する。一方、判定ブロック８０８がノー（偽）を返した場合、ステップ７２０は変数nextstripSPをヌルにセットする。

【００７７】ステップ８１０～８２０の完了後、ステップ７２０はステップ８２２へ進む。ステップ８２２中、変数origSPと、neighSPと、nextstripSPとに格納された位置のロケーションが比較され、これらの３つの位置から成る位置であって、ＪＰＥＧ２０００の走査順で第１となる当該位置が発見される。ステップ８２２の完了後、ステップ７２０はステップ８２４へ進み、そこでこの‘第１の’位置を含むエントリへのヘッド・ポインタは、対応するリスト内でこの第１のエントリから離れた方向へ１エントリだけ移される。比較ステップ８２２によって得られるこの‘第１の’位置は、次いで、さらなる処理を行うために、ＳＰ符号化ステップ３１４の符号化ステップ７２２とクリーンアップ・リスト生成ステップ７０８とへ渡される。次いで、ステップ７２０は終了する。

【００７８】ＳＰ符号化ステップ３１４がステップ７１０～７４０を実行中している間、ＳＰ符号化ステップ３１４はクリーンアップ・リスト生成ステップ７０８を実行する。これについて以下より詳細に説明する。

【００７９】次に図１３Ａと図１３Ｂに目を転じると、非有意近傍を新しいＳＰリストに追加する処理を示すフローチャートが示されている。ＳＰ符号化ステップ３１４は、ＳＰ符号化パス３１４中有意であることが判明した係数の、非有意近傍の（行、列）座標をステップ７３２（図７Ａ）中に渡す。これらの位置は、非有意近傍を新しいＳＰリストに追加するための処理１３００へ渡される。この処理１３００は、ＳＰ符号化ステップ３１４の開始から始まり、ＳＰ符号化ステップ３１４の完了で終了する。この処理１３００の目的は、有意係数の非有意近傍の任意のオーバーラップ部分を取り除くことである。例えば、２つの有意係数からなる非有意近傍である係数が存在するため、新しいＳＰリストへの追加を１回しか必要としない場合がある。これらの非有意近傍は、新しいＳＰリストに追加され、ＪＰＥＧ２０００の走査順にＳＰリスト内でソーティングが行われる。処理１３００は、４×Ｎの前のストリップ・レジスタと、４×Ｎ

の現在のストリップ・レジスタと、１×Ｎの前回のストリップ・ライン・ストアと、１×Ｎの次のストリップ・ライン・ストアと、前述した１×４の前回のストリップ・ライン・ストアとを利用する。但し、Ｎはブロックの幅である。処理１３００は、最初のクリーンアップ符号化パス３０８で用いられたような前処理ステップに多少類似した方法で前処理ステップとして機能する。処理１３００は、前回のストリップ・レジスタと前回のストリップ・ライン・ストアとの内容を後処理プロセス１４００へ送る。これについては図１４を参照してさらに詳細に後述する。この後処理プロセス１４００は、最初のクリーンアップ符号化パス３０８で用いられた方法と多少類似した方法で後処理ステップとして機能する。例えば、処理１３００がｉ番目のストリップを前処理している間、処理１４００は（ｉ－１）番目のストリップを後処理していることになる。

【００８０】前処理プロセス１３００は、ｉ番目のストリップ内の有意係数の３×３の近傍の範囲内に存在する非有意係数である、同じｉ番目のストリップ内の対応する非有意係数用の４×Ｎの現在のストリップ・レジスタのエントリのＮビットを（１）にセットする。前処理プロセス１３００はまた、（ｉ－１）番目のストリップの最後の行に在り、かつ、ｉ番目のストリップの第１の行の有意係数の３×３の近傍に在る非有意係数の列番号を前回のストリップ・ライン・ストア内に格納する。一旦、ｉ番目のストリップが前処理されると、該ｉ番目のストリップは後処理を行うために処理１４００へ送られる。処理１４００はまた前回のストリップ・ライン・ストア内のエントリであって、（ｉ＋１）番目のストリップの前処理中、前回のストリップ・ライン・ストアに現在追加されているエントリをフェッチする。即ち、後処理するプロセス１４００は、ｉ番目のストリップの最後の行内に在り、かつ（ｉ＋１）番目のストリップの第１の行の有意係数内の３×３の近傍に在る非有意係数の列番号を順にフェッチする。

【００８１】処理１３００はステップ１３０２から始まり、そこで、上述のレジスタのエントリとライン・ストアとはゼロにセットされる。次いで、処理１３００はステップ１３０４へ進み、そこでステップ７３２中に渡された有意係数の（行、列）座標がフェッチされる。このフェッチされた（行、列）座標は変数CurPosに格納される。次いで、処理１３００は判定ブロック１３０６へ進み、そこでCurPosに格納された、位置の３×３の近傍内に非有意係数が存在するかどうかのチェックが行われる。

【００８２】判定ブロック１３０６がイエス（真）を返した場合、処理１３００はループ１３１２～１３２４へ進み、そこで位置CurPosの３×３の近傍内の非有意近傍が順にフェッチされ、処理される。具体的には、処理１３００はステップ１３１２へ進み、そこで次の非有意近傍の（行、列）座標がフェッチされ、変数NextNに格納される。次いで、処理１３００は判定ブロック１３１４へ進み、そこで位置NextNが現在のストリップの前のストリップの中に存在するかどうかのチェックが行われる。判定ブロック１３１４がイエス（真）を返した場合、位置NextNの列番号が、前回のストリップ・ライン・ストアに追加される（１３１６）。イエス（真）を返さなかった場合、処理１３００は判定ブロック１３２０へ進み、そこで位置NextNが、現在のストリップの次のストリップの中に存在するかどうかのチェックが行われる。判定ブロック１３２０がイエス（真）を返した場合、位置NextNの列番号は次のストリップ・ライン・ストアに追加される（１３１８）。一方、判定ブロック１３２０がノー（偽）を返した場合、処理１３００は、位置NextNに対応する現在のレジスタ内のエントリに対してビットＮを（１）にセットする。ステップ１３１６、１３１８、又は１３２２の完了後、処理１３００は判定ブロック１３２４へ進む。判定ブロック１３２４で、非有意近傍がさらに存在するかどうかのチェックが行われる。判定ブロック１３２４がイエス（真）を返した場合、処理１３００はステップ１３１２へ戻り、次の非有意近傍のさらなる処理を行う。イエス（真）を返さなかった場合ステップは１３２８へ続く。

【００８３】ステップ１３２８で、変数PrevPosは変数CurPosにセットされる。次いで、処理１３００はステップ１３３０へ続き、そこで、処理１３００はステップ７３２（図７Ａ）中に渡された次の位置の（行、列）座標をフェッチする。このフェッチされた（行、列）座標は変数CurPosに格納される。この次の位置がまだ利用可能でない場合、処理１３００は利用可能になるまで待機する。それ以上利用可能な位置が存在しなければ、処理１３００はステップ１３３４へ直接進む（図示せず）。

【００８４】ステップ１３３０の完了後、処理は判定ブロック１３３２へ進み、そこで変数CurPos内に格納された位置が、変数PrevPosに格納された位置に関して、次のストリップ内に存在するかどうかのチェックが行われる。判定ブロック１３３２がノー（偽）を返した場合、処理１３００は判定ブロック１３０６へ戻り、変数CurPosの中に格納された新しい位置のさらなる処理が行われる。判定ブロック１３３２がイエス（真）を返した場合、処理１３００はステップ１３３４へ続く。

【００８５】ステップ１３３４中、処理１３００は現在のストリップ・レジスタの内容を前のストリップ・レジスタへ移動させ、現在のストリップ・レジスタ内のエントリをゼロにセットする。ステップ１３３４は、前回のストリップ・ライン・ストアへ次のストリップ・ライン・ストアの内容を移動させ、次いで、次のストリップ・ライン・ストア内のエントリをゼロにセットする。ステップ１３３４の完了後、処理はステップ１３３６と１３３８とへ同時に続く。ステップ１３３６中、処理１３０

０は、前のストリップ・レジスタを後処理する。これについては以下図１４を参照しながらさらに詳細に説明する。ステップ１３３８中、処理１３００は、前回のストリップ・ライン・ストア内に現在格納されている位置に対応する現在のストリップ・レジスタ内のエントリ用としてビットNを（1）にセットする。ステップ１３３６と１３３８との完了後、処理は、非有意近傍をさらに処理するために判定ブロック１３０６へ続く。

【００８６】判定ブロック１３０６がノー（偽）を返した場合、即ち、非有意近傍が存在しない場合、処理１３００は判定ブロック１３０８へ続く。判定ブロック１３０８は、ステップ７３２（図７Ａ）によって渡された位置がさらに存在するかどうかのチェックを行う。判定ブロック１３０８がイエス（真）を返した場合、処理１３００はステップ１３０４へ進み、そこでその位置はフェッチされる。現在利用可能な位置が存在しない場合には、判定ブロック１３０８は、位置が生じるまで待機するか、あるいは、ＳＰ符号化パス３１４は、処理が終了する（１３１０）終了時刻にＳＰ符号化パス３１４が終了するまで待機する。

【００８７】次に図１４に目を転じると、図１３Ａと１３Ｂの前処理１３００と共に使用される後処理１４００のフローチャートが示されている。後処理ステップ１４００はＳＰ符号化パス３１４と同時に始まる。後処理１４００は１４０２を開始し、ステップ１３３６中に後処理１４００へ渡される前のストリップ・レジスタを待機する（１４０４）。前のストリップ・レジスタが後処理１４００へ渡された後、処理１４００はループ１４０６～１４１６へ進む。ループ１４０６～１４１６は、前のストリップ・レジスタ内の各エントリ（即ち、位置）であって、（1）にセットされたNビットを持つ各エントリを順に処理し、前回のライン・ストア内に現在格納されているエントリ（列番号）とそのエントリとを比較し、ＪＰＥＧ２０００の走査順での以前の位置を新しいリストＳＰに追加する。ループ１４０６は、ＪＰＥＧ２０００の走査順に前のストリップ・レジスタ内のエントリを処理する。上記から明らかなように、前処理１３００は、エントリを追加することにより後処理１４００中も前回のライン・ストアを継続的に更新している。

【００８８】ループ１４０６～１４１６は判定ブロック１４０６とステップ１４０８とを有する。判定ブロック１４０６で、処理１４００は、（1）にセットされたNビットを持つ、前のストリップ・レジスタ内に何らかのエントリ（即ち、位置）が、または、前回のストリップ・ライン・ストア内に何らかの位置が存在するかどうかのチェックを行う。判定ブロック１４０６がイエス（真）を返した場合、処理１４００はステップ１４０８と１４１０へ同時に進む。ステップ１４０８で、処理１４００は、現在前回のライン・ストア内に在る次の（第1の）列番号をフェッチし、変数LinePosに該列番号を格納をする。前回のライン・ストア内に現在エントリが存在しなければ、変数LinePosはヌルにセットされる。同時に、ステップ１４１０は、前にフェッチされなかった次の位置であって、（1）にセットされたNビットを持つ次の位置を前のストリップ・レジスタからＪＰＥＧ２０００の走査順にフェッチし、変数StripPosにその位置を格納する。ステップ１４０８と１４１０の後、処理１４００はステップ１４１２へ進み、そこで処理は、変数LinePosとStripPos内に格納された位置の比較を行う。ステップ１４１２は、変数LinePosとStripPosに格納された該位置であって、ＪＰＥＧ２０００の走査順で早い位置にある該位置を決定し、変数WinPosにその位置を格納する。ステップ１４１２の完了後、処理はステップ１４１４へ進み、そこでその位置が前回のライン・ストア内に存在する場合、変数WinPosに格納された位置は、前回のライン・ストアから取り除かれる。ステップ１４１６の完了後、処理はステップ１４１６へ進み、そこで変数WinPosに格納された位置は新しいＳＰリストに追加される。処理１４００は更なる処理を行うために判定ブロック１４０６へ戻る。判定ブロック１４０６がノー（偽）を返した場合、処理１４００は終了する（１４１８）。

【００８９】次に図９に目を転じると、ＳＰ符号化ステップ３１４のクリーンアップ・リスト生成ステップ７０８のフローチャートが示されている。クリーンアップ・リスト生成ステップ７０８はステップ９０２から始まり、ステップ９０４へ進み、そこでループの第１のパス（７１０～７４０）中ステップ７２０で定められた位置を取得する。この位置がＪＰＥＧ２０００の走査順に従う第１の位置であり、この位置は、オリジナルのＳＰリスト、または近傍ＳＰリスト、または前回のストリップＳＰリストのいずれかの中に格納される。この位置は変数headSP内に格納される。また、ステップ９０４は、ＭＲリストに格納されている第１の位置をＪＰＥＧ２０００の走査順に従って取得し、変数headMRの中にこの位置を格納する。ステップ９０４の完了後、クリーンアップ生成ステップ７０８はループ９０６へ進む。

【００９０】ループ９０６は、ＪＰＥＧ２０００の走査順でコード・ブロックの各位置Ｐ(x,y)を通して１だけ増分する。コード・ブロックの新しい位置Ｐ(x,y)に対する各々の増分後、ループ９０６は判定ブロック９０８へ進む。判定ブロック９０８は、ループの現在位置Ｐ(x,y)が変数headSP内に格納された位置に等しいかどうかをチェックする。判定ブロックがノー（偽）を返した場合、クリーンアップ生成ステップ７０８は判定ブロック９１２へ進み、そこでループの現在位置Ｐ(x,y)が、変数headMRに格納された位置に等しいかどうかのチェックが行われる。判定ブロックがノー（偽）を返した場合、クリーンアップ生成ステップ７０８はステップ９１６へ進み、そこで現在位置Ｐ(x,y)がクリーンアップ・

リストに追加される。ステップ９１６の完了後、クリーンアップ生成ステップ７０８はループ９０６へ戻り、そこで、位置Ｐ(x,y)はＪＰＥＧ２０００の走査順で次の位置へ増分される。

【００９１】一方、判定ブロック９０８がイエス（真）を返した場合、クリーンアップ生成ステップ７０８はステップ９１０へ進む。ステップ９１０中、ステップ７２０で定められた次の位置が利用可能な場合、クリーンアップ生成ステップ７０８はその位置を取得し、変数headSPにその位置を格納する。このようにして、クリーンアップ生成ステップ７０８は、ステップ７２０のパス中に定められた位置を次々に検索し、ステップ９１０をコールする度に、変数headSPに格納された現在位置を次の位置と置き換える。次の位置が現在利用できない場合が時としてある。そのような状況では、クリーンアップ生成ステップ７０８はその位置が利用可能になるまで待機する。さらに、判定ブロック７１０（図７Ａ）がノー（偽）を返した場合、クリーンアップ生成ステップ７０８は、待機する代わりに、このステップ９１０中に、変数headSPをヌルにセットする。後者の場合、ＳＰ符号化ステップ中符号化の対象となる位置はそれ以上存在しない。しかし、ＳＰ符号化ステップ中、クリーンアップ・リストに追加する必要がある、符号化の対象となる位置が最後の位置の後に存在する可能性がある。変数headSPをヌルにセットすることにより、ループ９０６の後続パス中、これらの位置をクリーンアップ・リストに追加することができる。ステップ９１０の完了後、クリーンアップ生成ステップはループ９０６へ戻り、そこで、位置Ｐ(x,y)はＪＰＥＧ２０００の走査順で１だけ増分される。

【００９２】さらに、判定ブロック９１２がイエス（真）を返した場合、クリーンアップ生成ステップ７０８はステップ９１４へ進み、そこでＭＲリスト内の次の位置が変数headMRに格納される。このようにして、クリーンアップ生成ステップはＭＲリスト内の位置を次々に検索し、変数headMRに格納された現在位置をステップ９１４の各コールを用いて次の位置と置き換える。ＭＲリストに残された位置がそれ以上存在しなければ、ステップ９１２は変数headMRをヌルにセットする。ステップ９１４の完了後、クリーンアップ生成ステップはループ９０６へ戻り、位置Ｐ(x,y)は、ＪＰＥＧ２０００の走査順で１だけ増分される。

【００９３】コード・ブロックの範囲内のすべての位置Ｐ(x,y)がクリーンアップ生成ステップ７０８によって処理された後クリーンアップ生成ステップ７０８は終了する。

【００９４】このようにして、現在位置が現在のＳＰパス内で処理されている間、その位置はクリーンアップ・リスト生成ステップ７０８へも送られる。このステップ７０８では、ＪＰＥＧ２０００の走査順にブロック内のすべての座標を通るループが存在する。生成された座標は現在のＳＰ符号化パス内の次の位置と比較される。上記座標と位置とが等しければ、その位置は現在のパス内で予め処理され、その処理が続けられて現在のＳＰ符号化パス内に次の位置が取得され、その処理は次の座標を生成する。現在のＳＰ符号化パスから得られる次の位置が存在しなければ、位置が生じるまで、或いは、ＳＰ符号化が完了するまで処理は待機する。現在の座標が変数headSPに等しくない場合、現在の座標は変数headSPに格納されている座標より前の位置に存在し、したがってこの座標はクリーンアップ・リストの候補である。次いで、この座標は変数headMRと比較される。上記座標と位置とが等しくない場合、その座標は既にＭＲリストに在るため、処理はＭＲリスト内の次の位置を取得し、変数headMR内にそれを格納し、次の座標を生成する。この生成された座標が変数headMRに等しくない場合、この座標はクリーンアップ・リストの中へ挿入される。一旦、全ての位置が符号化されてしまうと、変数headSPをヌルにセットすることができる。同様に、いったんＭＲリストの全てのＭＲ位置が処理されてしまった場合、変数headMRはヌルにセットすることができる。このようにして、ＳＰリストとＭＲリストの最後の後の座標はクリーンアップ・リストに追加することができる。

【００９５】本構成の変更例では、処理が、ストリップ内の或る列の４つの位置がすべてクリーンアップ・リストに在ることを検出した場合、処理はクリーンアップ・リストの中で１つの合成座標にその４つの位置を圧縮することができる。この合成座標は、符号化が可能な場合、後の符号化／復号化のクリーンアップ段階で利用することができる。

【００９６】図７Ａに戻って、終了ステップ７１４中、ＳＰ符号化ステップ３１４はクリーンアップ符号化パス３１８中に新しいＳＰリストと更新されたオリジナルのＳＰリストとを返す。一方、エントロピ符号化法３００はマグニチュード・リファインメント符号化パス３１６へ進む。

【００９７】次に図１０に目を転じると、マグニチュード・リファインメント符号化パス３１６のフローチャートがさらに詳細に示されている。マグニチュード・リファインメント符号化パス３１６はステップ１００２から始まり、ループ１００４～１００８へ進み、そこでループ１００４～１００８のそれぞれのパス用のＭＲリスト内の位置がＪＰＥＧ２０００の走査順で検索される。ステップ１００２後、マグニチュード・リファインメント符号化パス３１６は、ＭＲリスト内の第１の位置の処理を行うためにループのステップ１００４と１００６とへまず進む。ステップ１００６の完了後、マグニチュード・リファインメント符号化パス３１６は判定ブロック１００８へ進み、そこで前に処理されなかった、処理の対象となる位置がＭＲリストの中にさらに存在するかどう

かのチェックが行われる。判定ブロック１００８がイエス（真）を返した場合、マグニチュード・リファインメント符号化パス３１６はステップ１００４へ戻り、そこで、MRリストの次の位置が検索される。ステップ１００４中、マグニチュード・リファインメント符号化パス３１６は、MRリストから現在位置をフェッチし、この現在位置を用いてメモリから対応するシンボルをフェッチする。マグニチュード・リファインメント符号化パス３１６はまた、現在位置の３×３の近傍内の有意値を有効値テーブルからフェッチする。ステップ１００４の完了後、マグニチュード・リファインメント符号化パス３１６はステップ１００６へ進む。ステップ１００６中、これらの有効値はコンテキスト生成処理へ送出されてコンテキストが生成され、対応するシンボルの符号化が行われ、さらに、そのシンボルに対して算術符号化が行われる。これらのステップ１００４と１００６とは１サイクル当たり１つの位置のレートで行うことができる。符号化処理では、ビットプレーン・メモリから必要なデータ・ビットをフェッチするために位置の座標が利用される。復号化処理では、出力用バッファの中に復号ビットを書き込むために位置の座標が利用される。判定ブロック１００８がノー（偽）を返した場合、マグニチュード・リファインメント符号化パス３１６は終了し（１０１０）、次いで、エントロピ符号化法はクリーンアップ符号化パス３１８へ進む。

【００９８】次に図１１に目を転じると、クリーンアップ符号化パス３１８のフローチャートが示されている。クリーンアップ符号化パス３１８はステップ１１０２から始まる。このステップ１１０２中、前回のシグニフィカンス・プロパゲーション符号化パス３１４中生成されたクリーンアップ・リストは現在のクリーンアップ符号化パス３１８へ渡される。また、前回のクリーンアップ符号化パス３０８または３１８に中生成され、前回の符号化パス３１４中に更新されたオリジナルのＳＰリストは、クリーンアップ符号化パス３１８へ渡される。さらに、前回の符号化パス３１４中生成された新しいＳＰリストは、クリーンアップ符号化パス３１８へ渡される。開始ステップ１１０２中、クリーンアップ符号化パス３１８は、位置の（行、列）座標を格納するためのクリーンアップ－有意係数リストとクリーンアップ非有意近傍リストと呼ばれる２つのリストも初期化し、それらのリストのエントリをヌルにセットする。最後に、前回のクリーンアップ符号化パス３０８または３１８中に生成されたMRリストは現在のクリーンアップ符号化パス３１８へ渡される。

【００９９】開始ステップ１１０２後、クリーンアップ符号化パス３１８はループ１１０４～１１１４へ進み、そこでクリーンアップ・リスト内の位置は、ＪＰＥＧ２０００の走査順でループ１１０４～１１１４のそれぞれのパス中検索される。ステップ１１０２後、クリーンアップ符号化パス３１８は判定ブロック１１０４へ進み、そこで現在のクリーンアップ符号化パス３１８によって検索されなかった位置がクリーンアップ・リストの中にさらに存在するかどうかのチェックが行われる。判定ブロック１１０４がイエス（真）を返した場合、クリーンアップ符号化パス３１８はステップ１１０６へ進み、そこで前に検索されなかった、クリーンアップ・リスト内の次の位置がフェッチされる。これらの位置は、ＪＰＥＧ２０００の走査順でクリーンアップ・リストから検索される。この位置がフェッチされた後、クリーンアップ符号化パス３１８はステップ１１０８へ進む。ステップ１１０８中、クリーンアップ符号化パス３１８はこの現在位置を利用して、その位置のビットプレーン・メモリから対応するビット・シンボルをフェッチする。クリーンアップ符号化パス３１８は、現在位置の３×３の近傍の有効値テーブルから有効値もフェッチする。さらに、これらの有意値はコンテキスト生成処理へ送出され、現在のシンボルの符号化用コンテキストが生成される。次いで、この現在のシンボルは算術符号化される。

【０１００】ステップ１１０８の完了後、クリーンアップ符号化パス３１８は判定ブロック１１１０へ進み、そこで符号化された現在のビット・シンボルが有意であるかどうかのチェックが行われる。判定ブロック１１１０がノー（偽）を返した場合、クリーンアップ符号化パス３１８は、クリーンアップ・リスト内の次の位置（もし存在する場合）の処理を行うために判定ブロック１１０４へ戻る。一方、判定ブロック１１１０がイエス（真）を返した場合、クリーンアップ符号化パスはステップ１１１２へ進み、そこで現在位置の（行、列）座標がクリーンアップ－有意係数リストに追加される。クリーンアップ－有意係数リスト内の位置はＪＰＥＧ２０００の走査順に格納される。ステップ１１１２の完了後、クリーンアップ符号化パス３１８はステップ１１１４へ進む。

【０１０１】ステップ１１１４中、クリーンアップ符号化パス３１８は、現在位置の３×３の近傍の範囲内の非有意係数を決定し、これらの非有意近傍の位置をクリーンアップ非有意近傍リストに追加する。クリーンアップ非有意近傍リスト内の後者の位置はＪＰＥＧ２０００の走査順に従ってソートされる。当業者には明らかなように、クリーンアップ非有意近傍リストは、クリーンアップ・リスト自身にない係数の位置を有することができる。これらの非有意係数をクリーンアップ非有意近傍リストに追加する方法は、新しいＳＰリストへの非有意係数の追加と関連して説明した方法（図１３Ａ、１３Ｂ、１４）と類似している。前者の場合、クリーンアップ符号化パス３１８は、図１３Ａ、１３Ｂ、１４を参照しながら説明したものと同じ処理を利用するが、以下の変更を伴う。クリーンアップ符号化パス３１８によって現在符号化されている有意係数の（行、列）座標は、ステップ１１１４中ステップ１３０４（図１３Ａ）へ渡され、

代わりにステップ１４１４で変数WinPosがクリーンアップ非有意近傍リストに追加される。このようにして非有意係数のいずれのオーバーラップも取り除かれる。

【０１０２】ステップ１１１４の完了後、クリーンアップ符号化パス３１８はＪＰＥＧ２０００に従って符号ビットを符号化し（図示せず）、次いで、判定ブロック１１０４へ戻る。判定ブロック１１０４がノー（偽）を返した場合、即ち、検索され、処理されたクリーンアップ・リストに位置がそれ以上存在しない場合、クリーンアップ符号化パス３１８はステップ１１１６へ進み、そこでクリーンアップ符号化パス３１８は終了する。

【０１０３】クリーンアップ符号化パス３１８がクリーンアップ・リスト内の各位置の処理を行っている間のと同時に、クリーンアップ符号化パスは、次のＳＰ符号化パス３１４とMR符号化パス３１６中、後続する処理を行うために、新しいオリジナルのＳＰリストと新しいMRリストとを生成する（１１１６）。

【０１０４】次に図１２に目を転じると、新しいオリジナルのＳＰリストと新しいMRリストとを生成するステップ１１１６のフローチャートがさらに詳細に示されている。ステップ１１１６は１２０２から始まり、そこで、更新されたオリジナルのＳＰリストと、新しいＳＰリストと、オリジナルのMRリストと、クリーンアップ－有意係数リストと、クリーンアップ非有意近傍リストとが入力される。リスト・メモリ内のオリジナルのＳＰリストは、前回の符号化ステップ３１４中更新された有意タグを用いてソートされ（１２０４）、２つのリストに変えられる。オリジナルのＳＰ有意係数リストは、オリジナルのＳＰリストの位置であって、現在のビットプレーン内の対応する有意係数を持つものとしてタグが付けられた位置を有する。後者の位置は新しいオリジナルMRリストに追加される。オリジナルのＳＰ非有意係数リストは、オリジナルのＳＰリストの位置を有する。該位置のタグは、現在のビットプレーン内の対応する非有意係数を持つ位置を示す。この後者の位置は、新しいオリジナルのＳＰリストに追加される。同様に、新しいＳＰリストは、前回の符号化ステップ３１４中に生成された有意タグを用いてソートされ（１２０６）、２つのリストに変えられる。新しいＳＰ有意係数リストは、新しいＳＰリストの位置であって、現在のビットプレーン内の対応する有意係数を持つものとしてタグが付けられた位置を有する。後者の位置は、新しいMRリストに追加される。新しいＳＰリストの位置を有する新しいＳＰ非有意係数リストであって、これらの位置を示すタグを持つオリジナルのＳＰ非有意係数リストは、現在のビットプレーン内の対応する非有意係数を持つ。後者の位置は新しいＳＰリストに追加される。

【０１０５】オリジナルのＳＰ－非有意係数リストの位置と、オリジナルのＳＰ－有意係数リストと、新しいＳＰ－非有意係数リストと、新しいＳＰ－有意係数リストと、オリジナルのMRリストと、クリーンアップ－有意係数リストと、クリーンアップ－非有意近傍リストとは、すべてのサブリストを一緒に併合する、２つの同一の‘比較と併合’処理１２０８と１２１０の中を通る。第１の‘比較と併合’処理１２０８は、オリジナルのＳＰ－有意係数リストの位置と、新しいＳＰ－有意係数リストと、オリジナルのMRリストと、クリーンアップ－有意係数リストとを比較し併合する（本明細書では、以後第１サブリストと呼ぶ）。第２の‘比較と併合’処理１２１０は、オリジナルＳＰ－非有意係数リストの位置と、新しいＳＰ－非有意係数リストと、クリーンアップ－非有意近傍リストとを比較し併合する（本明細書では、以後第２サブリストと呼ぶ）。‘比較と併合’処理１２０８と１２１０とは、クリーンアップ符号化ステップ３１８のループ１１０４～１１１４によって現在処理されているクリーンアップ・リストの現在位置も入力として受け取る。

【０１０６】‘比較と併合’処理１２１０は、第２のサブリストの中の各１つから１つの位置を受け入れ、それらを比較して、どれがＪＰＥＧ２０００の走査順で最も早い位置に在るかを調べる。次いで、‘比較と併合’処理１２１０は、ステップ１２１１へその結果を出力し、次いで、選択されたサブリストから次の位置を受け入れる。２つのサブリストが同じ位置を持っている場合、双方とも取り除かれ、１つだけが出力される。‘比較と併合’処理１２０８は、‘比較と併合’処理１２１０と同様に実行されるが、第１のサブリストからその位置を受け入れる。このソートティング処理は、１クロック・サイクル当たりのリスト当たり少なくとも１つの位置を生成することでき、したがってスループットは、サイクル当たり１係数／ビットのままであることが望ましい。しかし、ループ１１０４～１１１４によって処理される現在の座標が、これら全ての‘比較と併合’処理が出力しようとしている座標より前に在る場合、この‘比較と併合’処理は機能が停止する。この処理によって、新しいオリジナルＳＰとMRリスト内の座標の欠落が防止される。

【０１０７】実際には有意である非有意係数の位置が、第２のサブリストに存在する場合もある。これらの位置は、ステップ１２１１中、比較と併合処理１２１０の出力から取り除かれる。このステップ１２１１中、比較と併合処理１２１０によって出力される各位置は、有効値テーブル１２１３内のその位置における対応する係数の現在の有意状態と比較される。当該係数が有意であることが有効値テーブル１２１３によって示された場合、仮定上非有意係数の位置は、ステップ１２１１において取り除かれる。一方、当該係数が非有意であることが有効値テーブル１２１３によって示された場合、比較と併合処理１２１０によって出力された位置は取り除かれず、新しいオリジナルのＳＰリストの一部を形成する。

【０１０８】比較有意ステップ１２１１と比較と併合処理１２０８とは、新しいオリジナルのＳＰリストと、新しいオリジナルのＭＲリストとをそれぞれ出力し、これらのリストは後続するＳＰ符号化パス３１４とＭＲ符号化パス３１６で使用される。新しいオリジナルＳＰリストとＭＲリストとを生成するステップ１１１６は、リストの完了後、１２１２で終了する。

【０１０９】本構成の変更例では、新しいオリジナルのＳＰ／ＭＲリスト生成処理は継続し、次のビットプレーンのＳＰ符号化パス３１４に入る。次のＳＰ符号化パス３１４が始まるときまでに、新しいオリジナルのＳＰリストの先頭が準備ができていることが望ましい。それにより機能停止が不要となる。しかし、クリーンアップ符号化パス処理に起因する機能停止が生じないので、リスト生成処理は最大レートで進むはずである。

【０１１０】本構成のさらなる変更例では、オリジナルのＳＰリストと新しいＳＰリストのタグなしで済ますことが可能となる。現在のビットプレーンで有意であるかどうかを示す有効値テーブルから、これらのリストに格納された位置に対応する係数の有効値を読み出すことにより、リスト生成ステップ１１１６中、これらのリストをソートすることができる。更なる変更例では、リスト・メモリ内に、ビットが有意か否かを示す１ビットを設けて、その係数が以前に処理されたものであれば、このビットが書き込まれる。有効値テーブルへのアクセスを減らすという理由でこのアプローチは好適である。上記ビットはリスト・メモリとして同じメモリ内に在ってもよいし、或いは、別個のメモリ内に在ってもよい。

【０１１１】上述のように、クリーンアップ符号化パス３１８の完了後、本方法は任意のさらなるビットプレーンの処理を行うために判定ブロック３１０へ戻る。判定ブロックが真（イエス）を返した場合、即ち、処理の対象となる更なるビットプレーンが存在しない場合、本方法は終了する（３００）。

【０１１２】次に図１５に目を転じると、図３に図示のエントロピ符号化法を実行するためのエントロピ・コーデックが示されている。このエントロピ・コーデックは、リスト・メモリ・マネージャ１５０２と、ビットプレーン・スプリッタ１５０４と、ビットプレーン・メモリ１５０８と、有効値テーブル１５１０と、リスト・メモリ１５０６と、コンテキスト生成論理回路１５１２と、算術符号化部（図示せず）とを有する。リスト・メモリ・マネージャ１５０２は、専用集積回路の形をとり、その機能は図３のエントロピ符号化法の演算と関連づけられるリスト作成を実現することである。リスト・メモリ・マネージャ１５０２によって作成されたこれらのリスト（例えば新しいＳＰリスト）は、リスト・メモリ・マネージャ１５０２によって、半導体メモリの形のリスト・メモリ１５０６内に格納される。リスト・メモリ・マネージャ１５０２は、図３のエントロピ符号化法自体のコンテキスト生成ステップと符号化ステップとを実行せず、コンテキスト生成論理回路１５１２とコーデック（図示せず）によって行われる、これらの操作の制御を行う。同様に、リスト・メモリ・マネージャ１５０２は、ビットプレーン分割操作３０４と操作３０６とを行わず、ビットプレーン・スプリッタ１５０４によって実行される、これらの操作の制御を行う。

【０１１３】コーデック１５００は以下のように図３のエントロピ符号化法を実行する。コード・ブロックのウェーブレット係数がＤＷＴユニット（図示せず）から入力されている間、このコード・ブロックを構成するビットプレーンは、ビットプレーン・スプリッタ１５０４によってそれぞれのビットプレーン・メモリ１５０８の中へ書き込まれる。同時に、ビットプレーン・スプリッタ１５０４は、各係数が有意となり始めるビットプレーンを決定し、有効値テーブル１５１０内にその情報を格納する。また同時に、ビットプレーン・スプリッタ１５０４は、どれがビットプレーン内で有効ビットを持つ最上位ビットプレーンであるかを決定し、リスト・メモリ・マネージャ１５０２へ最上位ビットプレーン番号を送出する。リスト・メモリ・マネージャ１５０２はまた、符号化の対象となる最下位ビットプレーンのビットプレーン番号もフェッチする。リスト・メモリ・マネージャ１５０２は、最上位ビットプレーンから始まり、次いで、各ビットプレーンについて、シグニフィカンス・プロパゲーション・パス３１４のオペレーションと、マグニチュード・リファインメント・パス３１６と、クリーンアップ・パス３０８または３１８とを符号化の対象となる最下位ビットプレーンまで実行する。各パスで、リスト・メモリ・マネージャ１５０２は、そのパスで符号化／復号を行う必要がある全ての係数である、リスト・メモリ１３０６の中に格納されている全ての係数の位置リストを作成する。シグニフィカンス・プロパゲーション・パス３１４の場合、リスト・メモリ・マネージャ１５０２がオリジナルのＳＰリストの中を通るとき、新しいＳＰリストが生成される。

【０１１４】リスト・メモリ・マネージャ１５０２は、符号化の対象となるシンボルの位置を決定するとすぐに、有効値テーブル１５１０から読み出してコンテキスト生成論理回路１５１２上へ渡すべきその位置の３×３の近傍内の係数の有意値を指示する。リスト・メモリ・マネージャ１５０２はまた、現在のビットプレーン内のその位置におけるシンボルをビットプレーン・メモリ１５０８から検索する。次いで、コンテキスト生成論理回路１５１２は、この有効値情報に基づいてコンテキストを計算し、算術符号化部（図示せず）へその情報を渡す。また、現在のビットプレーン内のその位置におけるシンボルは算術符号化部（図示せず）上へ渡される。

【０１１５】次に図１６に目を転じると、コード・ブロックの変換係数のシンボルのエントロピ符号化を行う別

の方法１６００を示すフローチャートが示されている。シンボルのエントロピ符号化を行うこの方法１６００は、図３を参照しながら説明したシンボルの符号化を行う方法より単純である。しかし、この方法１６００は係数のうちの数ビットを予め知っている必要があるので、エントロピ復号化を行うために、この方法を用いることはできない。したがって、この方法は、単純ではあるものの、エントロピ符号化時にしか利用することができない。

【０１１６】本方法１６００はステップ１６０２から始まり、このステップで必要なパラメータが全て初期化される。初期化段階１６０２中、本方法１６００は有効値テーブルを生成する。この有効値テーブルは、符号化の対象となるコード・ブロックの係数の有意状態の２次元配列を有する。本方法１６００は、コード・ブロック全体を処理し、各係数が有意となり始める位置に在るビットプレーンを決定し、有効値テーブル内にその情報を格納する。次いで、本方法１６００は、その係数と関連する有効値テーブルに格納されるビットプレーン番号に関して、現在のビットプレーンにおける係数の有意状態を決定することができる。例えば、有効値テーブル内に、ある係数と関連づけられて格納されている現在のビットプレーンがビットプレーン番号より大きい場合、その係数はその現在のビットプレーンについて非有意となる。現在のビットプレーンがビットプレーン番号より大きくない場合にはその係数は有意になる。このビットプレーン番号は、コード・ブロックの範囲内の係数位置に対応する配列を示す（行、列）座標と共に、配列として有効値テーブルの中に格納される。

【０１１７】また初期化段階１６０２中、本方法は、シグニフィカンス・プロパゲーションリスト（ＳＰリスト）と、マグニチュード・リファインメント・リスト（ＭＲリスト）と、クリーンアップ・リスト（Ｎリスト）として本明細書で名付けられた３つのリストを生成する。これらのリストには、現在のパス中、符号化されるコード・ブロックの範囲内のビット・シンボルの位置が含まれる。例えば、このシグニフィカンス・プロパゲーションリストには、現在のシグニフィカンス・プロパゲーション・パスで符号化されるコード・ブロックの範囲内の全てのビット・シンボルの位置リストが含まれる。‘ビット・シンボルの位置’という用語は、そのビット・シンボルが一部を形成する変換係数のコード・ブロックの範囲内の配列位置を、本明細書では意味する。さらに、これらのリストの各々の範囲内の位置は、ＪＰＥＧ２０００の走査順に従って索引化される（図２参照）。最初、シグニフィカンス・プロパゲーションリスト（ＳＰリスト）とマグニチュード・リファインメント・リスト（ＭＲリスト）とは空である。一方、クリーンアップ・リスト（Ｎリスト）は、コード・ブロックの範囲内の全ての位置をリストし、ノンゼロ・ビットを持つ最上位ビットプレーンの符号化を行うために使用される。

【０１１８】開始１６０２後、コード・ブロックの変換係数が受け取られ、ビットプレーンに分割される（１６０４）。次のステップ１６０６では、コード・ブロックのビットプレーンがチェックされ、どのビットプレーンが最上位ノンゼロ・ビットを所有するかが判定される。

【０１１９】次いで、本方法はステップ１６０８へ進み、そこで最上位ビットを持つビットプレーンは、１回の単一クリーンアップ・パスでエントロピ符号化される。具体的には、最上位ノンゼロ・ビットを持つビットプレーンに属するビット・シンボルが各々エントロピ符号化される。これらのビット・シンボルは、例えば、図２に図示のように、ＪＰＥＧ２０００の走査順でエントロピ符号化される。エントロピ符号化ステップ１６０８は、符号化対象ビット・シンボルと、そのビット・シンボルのコンテキストとを入力として受け取り、該コンテキストはＪＰＥＧ２０００に従って好適に決定される。

【０１２０】ステップ１６０８の完了後、本方法は判定ブロック１６１０へ進み、そこで処理の対象となるビットプレーンがさらに存在するかどうかのチェックが行われる。もし存在すれば、本方法はステップ１６１２へ進み、そこで次のビットプレーンのビット・シンボルが検索される。

【０１２１】ステップ１６１２の完了後、本方法１６００はステップ１６１３へ進む。ステップ１６１３中、本方法１６００は、まずシグニフィカンス・プロパゲーションリスト（ＳＰリスト）と、マグニチュード・リファインメント・リスト（ＭＲリスト）と、クリーンアップ・リスト（Ｎリスト）とをクリアする。その後、ステップ１６１３では、コード・ブロックの範囲内のすべての位置をソートし、シグニフィカンス・プロパゲーションリスト（ＳＰリスト）と、マグニチュード・リファインメント・リスト（ＭＲリスト）と、クリーンアップ・リスト（Ｎリスト）との中にこれらの位置を格納する。ソート・ステップ１６１３は、これらのリストのうちの１つだけに各位置が格納されるように、これらの位置を格納する。これらの格納された位置は、コード・ブロック内の対応する係数の（行、列）座標を示す。

【０１２２】ソート・ステップ１６１３中、本方法はＪＰＥＧ２０００の走査順にコード・ブロックの全ての位置に亙って増分を行う。

【０１２３】現在の位置走査中、ソート・ステップ１６１３は、現在走査中の位置を取り囲む３×３の近傍の位置に対応する、有効値テーブルに格納されたビットプレーン番号を検索する。ソート・ステップはまた、現在走査中の位置に対応する有効値テーブル内に格納されたビットプレーン番号も検索する。上述のように、有効値テーブルに格納されたビットプレーン番号は、その係数が有意になる位置に存在するビットプレーンを示す。次い

でソート・ステップ１６１３は、現在位置の係数の有意状態を含む、３×３の近傍の各係数の有意状態を、現在のビットプレーンにおいて決定する。次いで、ソート・ステップ１６１３は、以下の規則に従って、ＳＰ、ＭＲまたはＮリストのうちのいずれに、その位置を格納すべきかを判定する。

【０１２４】規則１：走査中の現在位置Ｐ(x,y)の係数が有意状態≧Ｎ（但し、Ｎは処理中の現在のビットプレーン）を持っていれば、現在位置はＭＲリストに進む。

【０１２５】規則２：走査中の現在位置Ｐ(x,y)の係数が、Ｎより小さい（＜Ｎ）有意状態を持ち、かつ、ＪＰＥＧ２０００の走査順で現在位置Ｐ(x,y)の前に在る係数である、現在位置Ｐ(x,y)の３×３の近傍内の任意の係数が、有意状態≧Ｎ（但し、Ｎは処理中の現在のビットプレーン）を持つ係数を持っていれば、現在位置Ｐ(x,y)はＳＰリストに入る。

【０１２６】規則３：走査中の現在位置Ｐ(x,y)の係数が、Ｎより小さい有意状態（＜Ｎ）を持ち、かつ、ＪＰＥＧ２０００の走査順で現在位置Ｐ(x,y)の後に在る現在位置Ｐ(x,y)の３×３の近傍内の任意の位置であって、有意状態＞Ｎ（但し、Ｎは処理中の現在のビットプレーン）を持つ位置を持っていれば、現在位置Ｐ(x,y)はＳＰリストに入る。

【０１２７】規則４：走査中の現在位置Ｐ(x,y)がＭＲリスト又はＳＰリストに格納されていなければ、その位置はＮリストに格納される。即ち、クリーンアップ（Ｎ）リストには、ＳＰリストとＭＲリストにない位置が含まれる。

【０１２８】このようにして、ソート・ステップ１６１３は、係数の位置をソートしてビットプレーンＮ用のＳＰ、ＭＲ、Ｎリストの中へ入れる。次いで、これらのリストを送って、シグニフィカンス・プロパゲーション・パスと、マグニチュード・リファインメント・パスと、クリーンアップ・パスとの中で符号化を行うことができる。

【０１２９】ソート・ステップ１６１３の完了後、本方法はステップ１６１４へ進み、現在のビットプレーンのシグニフィカンス・プロパゲーション・パス内で関連するビット・シンボルの処理が行われる。このステップ１６１４で、ステップ１６１３で生成されたビット・シンボルである、現在のＳＰリスト内の位置に対応する現在のビットプレーン内のビット・シンボルがエントロピ符号化される。

【０１３０】ＳＰ符号化パス１６１４の完了後、本方法はマグニチュード・リファインメント符号化パス１６１６へ進む。このステップ１６１６で、ステップ１６１３で生成されたビット・シンボルである、現在のマグニチュード・リファインメント・リスト内の位置に対応する現在のビットプレーン内の全てのビット・シンボルがエントロピ符号化される。

【０１３１】ＭＲ符号化パス１６１６の完了後、本方法はクリーンアップ符号化パス１６１８へ進む。クリーンアップ符号化ステップ１６１８で、（前回の符号化パス中生成されたビット・シンボルである）、現在のクリーンアップＮリスト内の位置に対応する、現在のビットプレーン内の全てのビット・シンボルがエントロピ符号化される。

【０１３２】クリーンアップ符号化パス１６１８の完了後、本方法は判定ブロック１６１０へ戻り、任意の更なるビットプレーンの処理が行われる。判定ブロックが真（イエス）を返した場合、即ち、処理の対象となる更なるビットプレーンが存在しない場合、本方法は終了し（１６２０）、コール方法へ戻る。判定ブロックが真（イエス）を返さなかった場合、本方法は、次のビットプレーンの処理を行うためにステップ１６１２へ進む。好適には、本方法は所定の最下位ビットプレーンまでビットプレーン処理を続けることが望ましい。

【０１３３】好適には、本方法が全ての３つのリスト、即ち、ＳＰ、ＭＲ及びＮリストを生成することが望ましい。本構成の変更例では、本方法はＳＰ、ＭＲ及びＮリストのうちの少なくとも１つのリストを生成することができる。この場合、本方法はリストを持っていない任意のパス中コード・ブロック全体を走査する。

【０１３４】次に図１７に目を転じると、図１６に図示のエントロピ符号化法を実行するエントロピ符号器が示されている。エントロピ符号器１７００は、図１６を参照しながら上述した規則に従って現在のビットプレーンの係数の位置をソートする係数ソータと、操作１６０４と１６０６とを行うためのビットプレーン・スプリッタ（図示せず）と、コード・ブロックの各ビットプレーンのビット・シンボルを格納するためのビットプレーン・メモリ（図示せず）と、有意値を格納するための有効値テーブル１７１０とを有する。エントロピ符号器はまた、係数ソータ１７０２によって生成されるＳＰリストを格納するためのＳＰリスト・メモリ１７０４と、係数ソータ１７０２によって生成されるＭＲリストを格納するためのＭＲリスト・メモリ１７０６と、係数ソータ１７０２によって生成されるクリーンアップ・リストを格納するためのクリーンアップ・リスト・メモリ１７０８をも有する。エントロピ符号器は、図１６を参照しながら説明したように、コンテキスト生成論理回路１７１２と、ＳＰリストと、ＭＲリストと、クリーンアップ・リスト内に、それらのリストのそれぞれのパス中格納された位置に対応するビット・シンボルを符号化するための算術符号化部（図示せず）とを更に有する。ＳＰリスト・メモリ１７０４は、ダブルバッファ構成である。係数ソータ１７０２が、ビットプレーンｎのＳＰリスト用の位置を生成して、それらの位置をＳＰリスト・メモリ１７０４内に格納している間、コンテキスト生成論理回路１７１２はビットプレーン（ｎ＋１）用の完成したＳＰ

リストを読み出している。ＭＲリスト・メモリ１７０６とクリーンアップ・リスト・メモリ１７０８とはまた、ＳＰリスト・メモリ１７０４と同様の方法でダブルバッファとしても機能する。

【０１３５】図３と図１６を参照しながら説明した前述の好ましい方法は、ある特別の制御フローを有する。これらの方法には、本発明の精神または範囲から逸脱することなく様々な制御フローを用いる他の多くの変形例が存在する。例えば、図３のエントロピ符号化法は、いくつかの並列操作を部分的に有する。図３のエントロピ符号化法を改変して、当業者には明らかであるようなシーケンシャルな操作であるかのように、これらの並列操作を実現することもできる。後者（シーケンシャルなオペレーション）の場合、図３のこの改変されたエントロピ符号化法は、汎用コンピュータのソフトウェアとしての実現に適している。例えば、前処理ステップ４１２と後処理ステップ４１８とは、いくつかの改変例についてはシーケンシャルに機能してもよい。汎用コンピュータのソフトウェアとして図１６のエントロピ符号化法を実現してもよい。

【０１３６】次に図１８に目を転じると、前述の方法を実行することができる汎用コンピュータが示されている。コンピュータ・システム１８００の範囲内で実行するアプリケーション・プログラムのようなソフトウェアとして、これらの方法の処理を実行してもよい。特に、前述の方法のステップは、コンピュータによって実行されるソフトウェア内の命令によって実行される。例えば、以下説明する記憶装置を含むコンピュータ可読媒体にこのソフトウェアを格納してもよい。このソフトウェアは、コンピュータ可読媒体からコンピュータの中へロードされ、次いで、コンピュータによって実行される。コンピュータ可読媒体に保存されるようなソフトウェアまたはコンピュータ・プログラムを持つコンピュータ可読媒体は、コンピュータ・プログラム製品である。

【０１３７】コンピュータ・システム１８００は、コンピュータ・モジュール１８０１と、キーボード１８０２とマウス１８０３等の入力装置と、プリンタ１８１５と表示装置１８１４とを含む出力装置とを有する。コンピュータ・モジュール１８０１は、電話回線１８２１またはその他の機能媒体を介して接続可能な通信ネットワーク１８２０と交信するための変復調（モデム）送受装置１８１６を使用する。モデム１８１６を使用して、インターネットやローカル・エリア・ネットワーク（ＬＡＮ）や広域ネットワーク（ＷＡＮ）等の他のネットワーク・システムにアクセスすることもできる。

【０１３８】コンピュータ・モジュール１８０１には、少なくとも１つのプロセッサ・ユニット１８０５と、例えば半導体ランダム・アクセス・メモリ（ＲＡＭ）やリード・オンリ・メモリ（ＲＯＭ）から形成される記憶装置（メモリ）１８０６と、ビデオ・インターフェース１８０７を含む入力／出力（Ｉ／Ｏ）インターフェースと、キーボード１８０２とマウス１８０３およびオプションとしてジョイ・スティック（例示せず）用のＩ／Ｏインターフェース１８１３と、モデム１８１６用のインターフェース１８０８とが一般に含まれる。記憶装置１８０９が提供され、この装置には一般にハードディスク・ドライブ１８１０とフロッピー（登録商標）ディスク・ドライブ１８１１とが含まれる。磁気テープ駆動装置（例示せず）を使用することもできる。不揮発性データソースとしてＣＤ－ＲＯＭ駆動装置１８１２が一般に設けられる。コンピュータ・モジュール１８０１の構成要素１８０５～１８１３は、一般に、相互接続バス１８０４を介して、また、コンピュータ・システム１８００の従来の動作モードで結果として得られる当業者に公知の方法で交信を行う。実施の形態の実行が可能なコンピュータの例には、ＩＢＭ－ＰＣおよび互換機、Sun Sparstationやそこから進化した同様のコンピュータ・システムが含まれる。

【０１３９】一般に、前述の方法のアプリケーション・プログラムはハードディスク・ドライブ１８１０に常駐し、プログラムの実行時にプロセッサ１８０５によって読み出され、制御される。半導体メモリ１８０６を用いて、おそらくハードディスク・ドライブ１８１０に従って、ネットワーク１８２０からフェッチされたプログラムと任意のデータの中間的格納を行うことができる。いくつかの例では、ＣＤ－ＲＯＭやフロッピーディスクに符号化され、対応する駆動装置１８１２や１８１１を介して読み出されるアプリケーション・プログラムや、或いは、ユーザーがネットワーク１８２０からモデム装置１８１６を介して読み出すことができるアプリケーション・プログラムをユーザへ供給することができる。さらに、磁気テープ、ＲＯＭあるいは集積回路、光磁気ディスク、コンピュータ・モジュール１８０１と別の装置との間の無線伝送チャネルまたは赤外線伝送チャネル、ＰＣＭＣＩＡカードのようなコンピュータ可読カード、ウェブ等に保存されるｅ－メール伝送と情報を含むインターネットとイントラネットを含む他のコンピュータ可読媒体からコンピュータ・システム１８００の中へソフトウェアをロードすることもできる。上述のものは単に関連するコンピュータ可読媒体の例示にすぎない。本発明の範囲と精神から逸脱することなく、他のコンピュータ可読媒体を実際に使用することも可能である。

【０１４０】本発明の実施の形態がコンピュータ産業とビデオ画像産業、並びに、ビデオ及びカメラ産業のような関連分野に適用可能であることは、以上のことから明らかである。

【０１４１】上述のことは、本発明のいくつかの実施の形態について説明するものにすぎず、本発明の範囲と精神から逸脱することなく改変及び／又は変更を行うことが可能であり、以上の実施の形態は例示的なものであっ

て、限定なものではない。

【図面の簡単な説明】

【図１】従来技術のＪＰＥＧ２０００で利用される係数“Ｘ[ｍ，ｎ]”の８個の包囲係数の近傍有意状態を示す図である。

【図２】従来技術のＪＰＥＧ２０００で利用されるコード・ブロック用のコード・ブロック走査パターンの一例を例示する概略ブロック図である。

【図３】第１の構成に従うエントロピ符号化法を示すフローチャートである。

【図４】図３に図示のエントロピ符号化法で用いられる最初のクリーンアップ符号化ステップ３０８の詳細処理を示すフローチャートである。

【図５】図４に描かれるようなｉ番目のストリップのｊ番目の列の前処理ステップ４１２をさらに詳細に示すフローチャートである。

【図６】図４に描かれるような（ｉ－１）番目のストリップの（ｊ－２）番目の列の後処理ステップ４１８をさらに詳細に示すフローチャートである。

【図７Ａ】、

【図７Ｂ】図３のシグニフィカンス・プロパゲーション（ＳＰ）符号化ステップ３７４をさらに詳細に示すフローチャートである。

【図８】ＳＰ符号化ステップ３１４のステップ７２０をさらに詳細に示すフローチャートである。

【図９】ＳＰ符号化ステップ３１４のクリーンアップ・＊リスト生成ステップ７０８をさらに詳細に示すフローチャートである。

【図１０】マグニチュード・リファインメント符号化パス３１６をさらに詳細に示すフローチャートである。

【図１１】図３のクリーンアップ符号化パス３１８をさらに詳細に示すフローチャートである。

【図１２】新しいオリジナルのＳＰリストと新しいＭＲリストとを生成するためのステップ１１１６をさらに詳細に示すフローチャートである。

【図１３Ａ】、

【図１３Ｂ】図７Ａのステップ７３２によって渡される新しいＳＰリストに非有意近傍を加える処理を示すフローチャートである。

【図１４】図１３Ａと図１３Ｂの前処理プロセスとともに使用する後処理プロセスを示すフローチャートである。

【図１５】図３に図示のようなエントロピ符号化法を実現するためのエントロピ・コーデックの概略図を示す図である。

【図１６】第２の構成に従うエントロピ符号化法を示すフローチャートである。

【図１７】図１６に示すエントロピ符号化法を実現するためのエントロピ符号器の概略図である。

【図１８】図３と図１６の方法を実行する汎用コンピュータの概略図である。

【図1】

Si[m-1,n-1]	Si[m-1,n]	Si[m-1,n+1]
Si[m,n-1]	"X[m,n]"	Si[m,n+1]
Si[m+1,n-1]	Si[m+1,n]	Si[m+1,n+1]

Fig. 1　Prior Art

【図2】

1	5	9	13	17	21	25	29
2	6	10	14	18	22	26	30
3	7	11	15	19	23	27	31
4	8	12	16	20	24	28	32
33	37	41	45	49	53	57	61
34	38	42	46	50	54	58	62
35	39	43	47	51	55	59	63
36	40	44	48	52	56	60	64

Fig. 2　Prior Art

【図3】

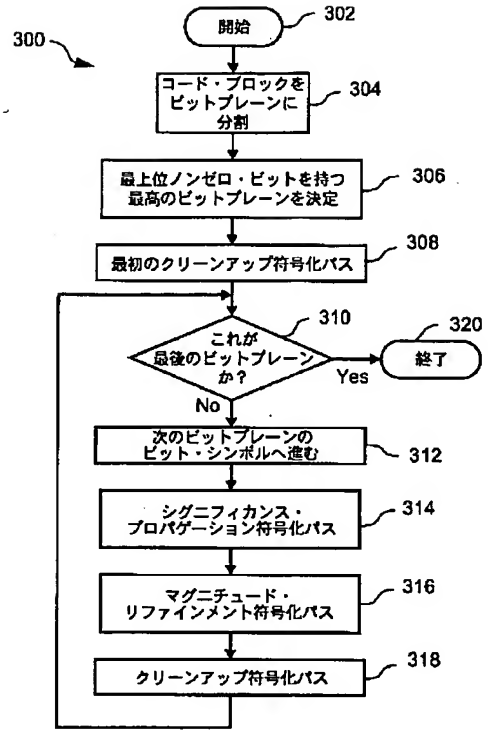


Fig. 3

【図4】

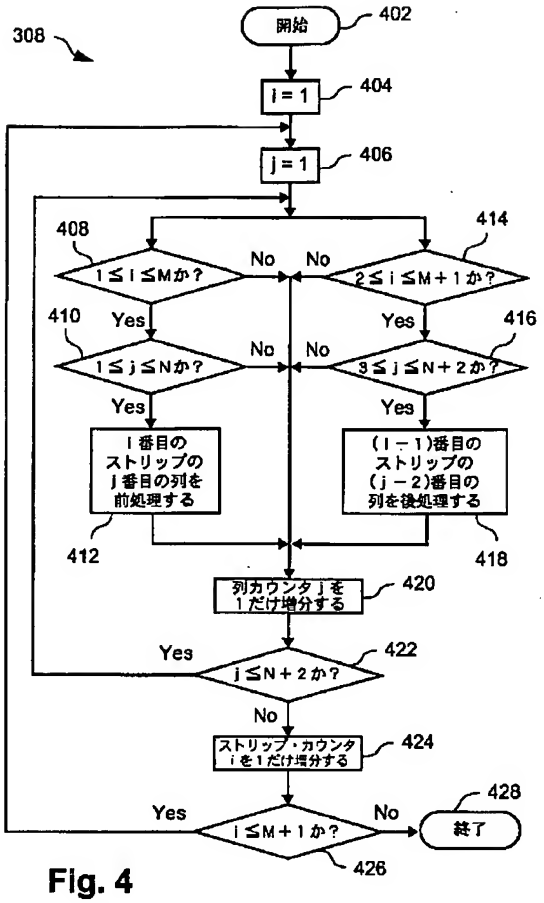


Fig. 4

【図8】

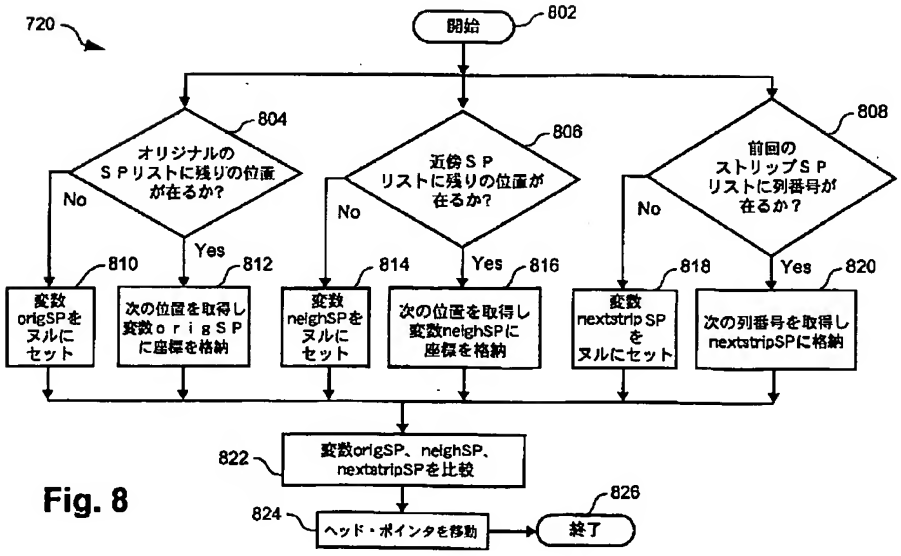


Fig. 8

【図５】

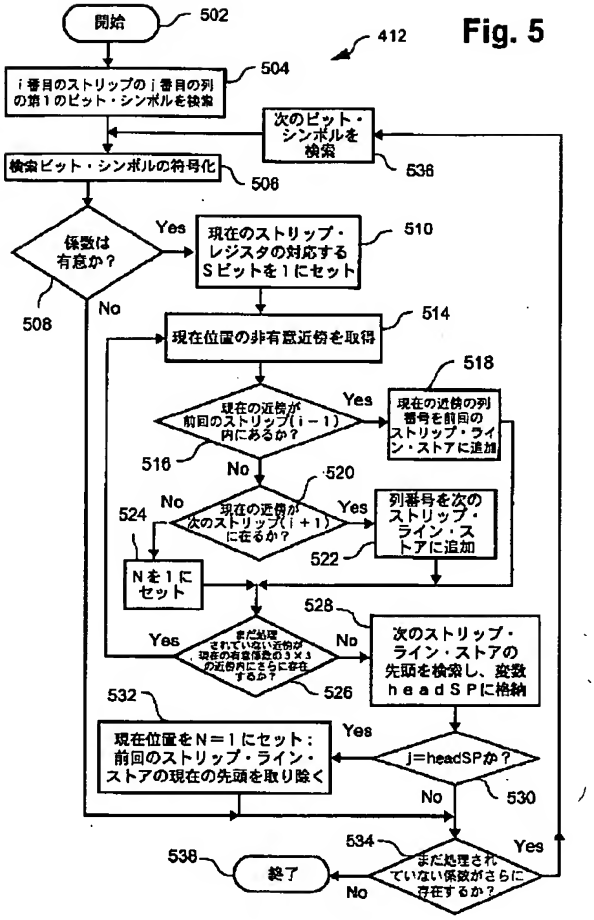


Fig. 5

【図６】

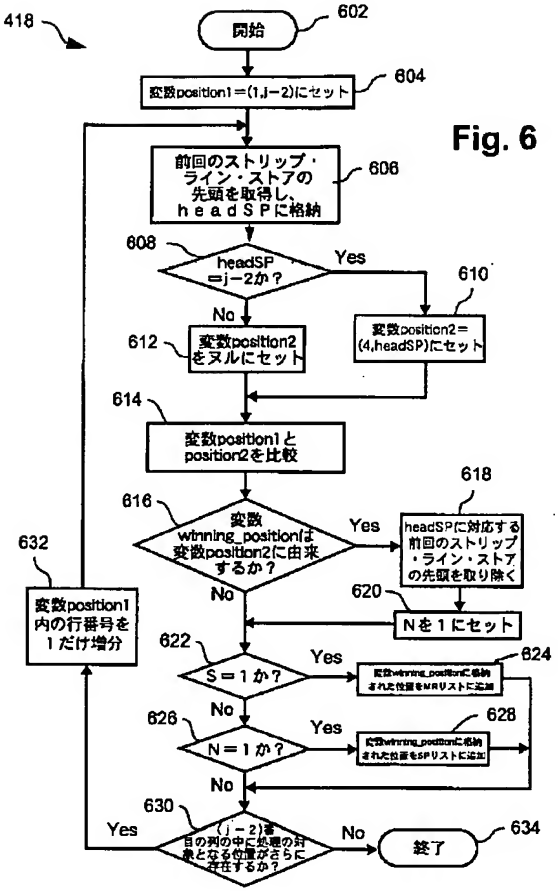


Fig. 6

【図１５】

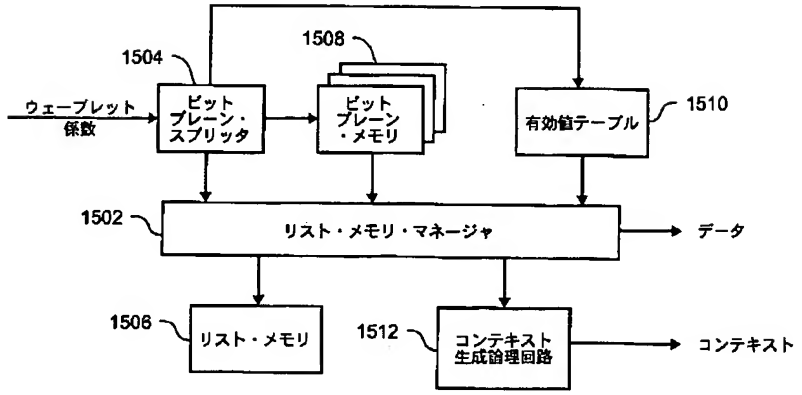
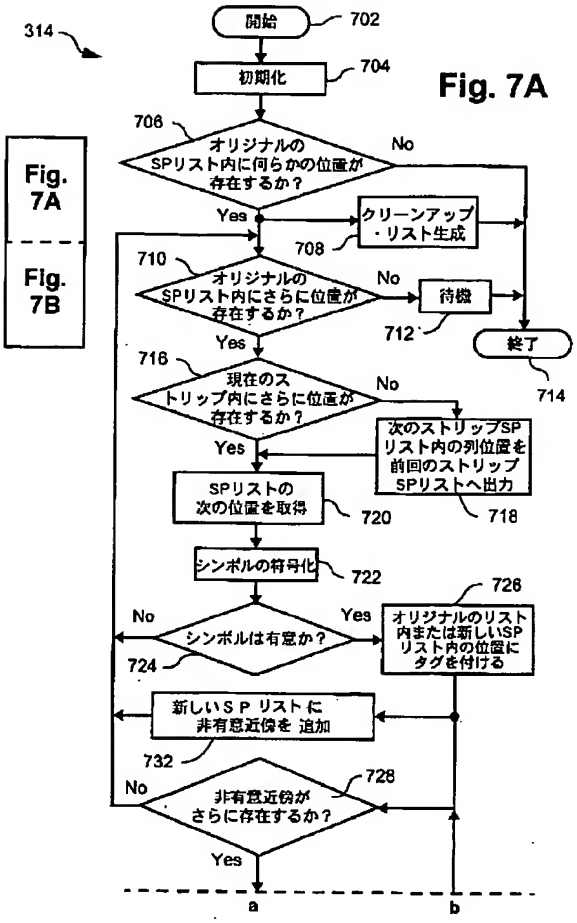
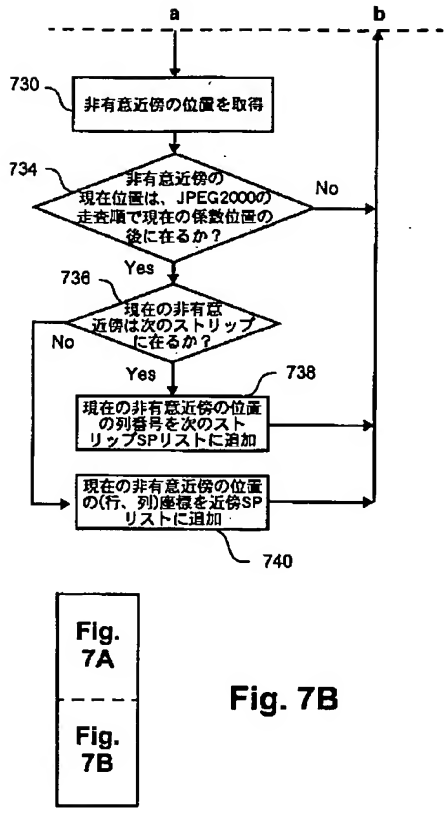


Fig. 15

【図７Ａ】



【図７Ｂ】



【図１０】

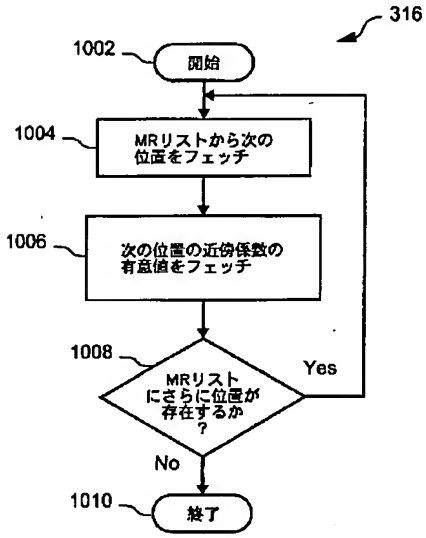


Fig. 10

【図９】

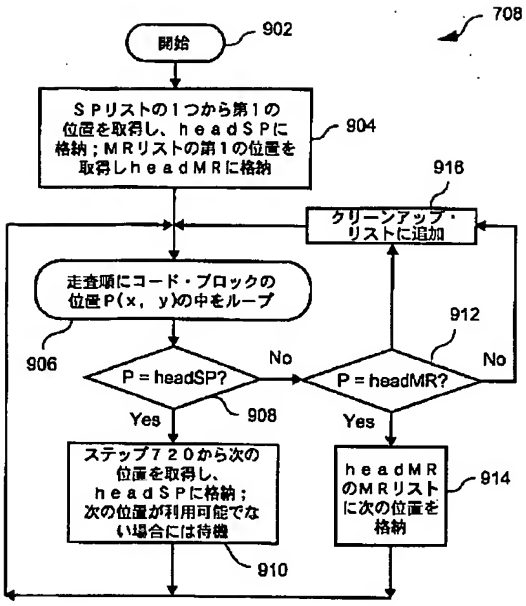


Fig. 9

【図１１】

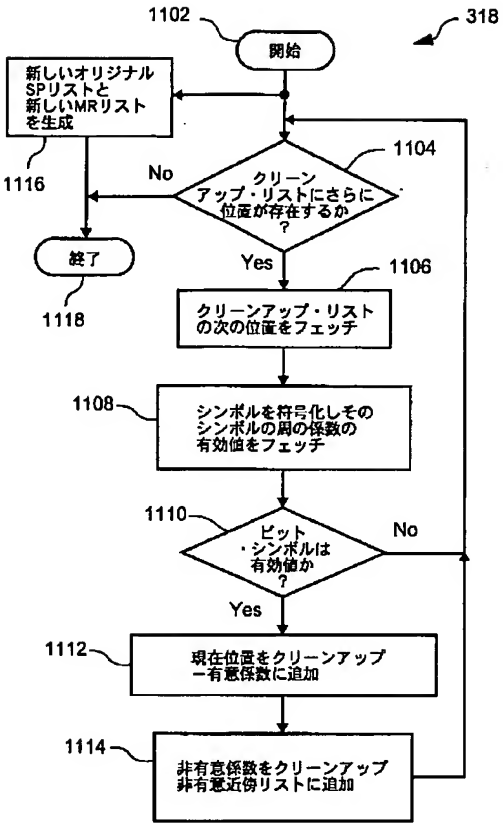


Fig. 11

【図１２】

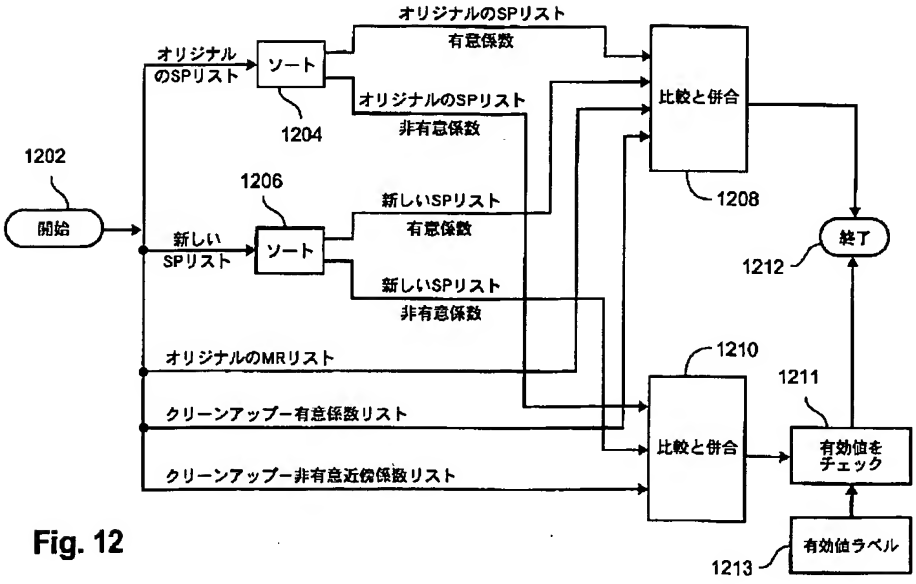
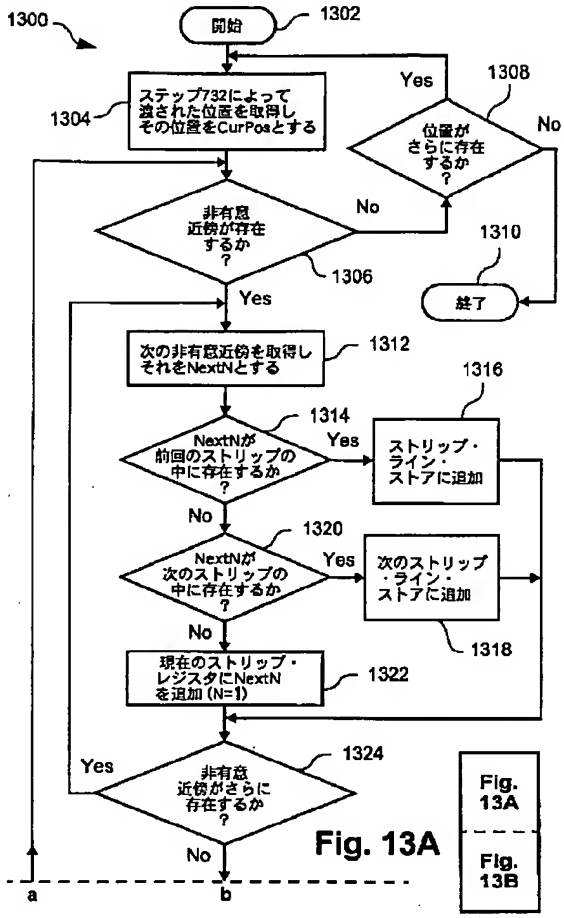


Fig. 12

【図１３Ａ】



【図１３Ｂ】

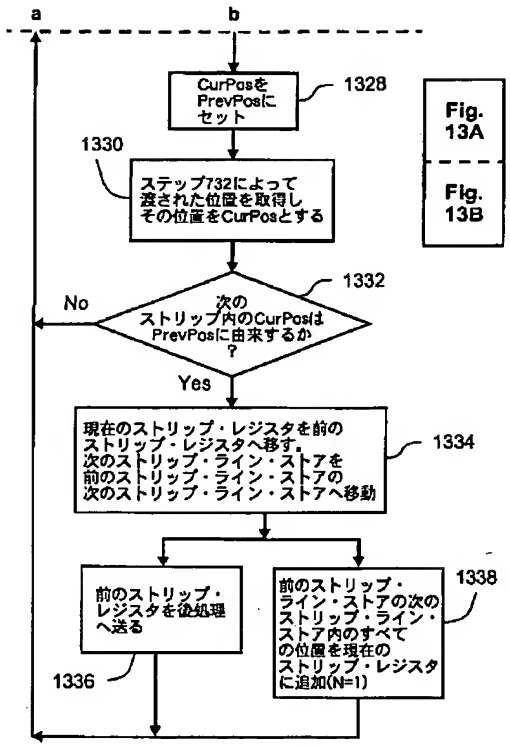


Fig. 13B

【図１７】

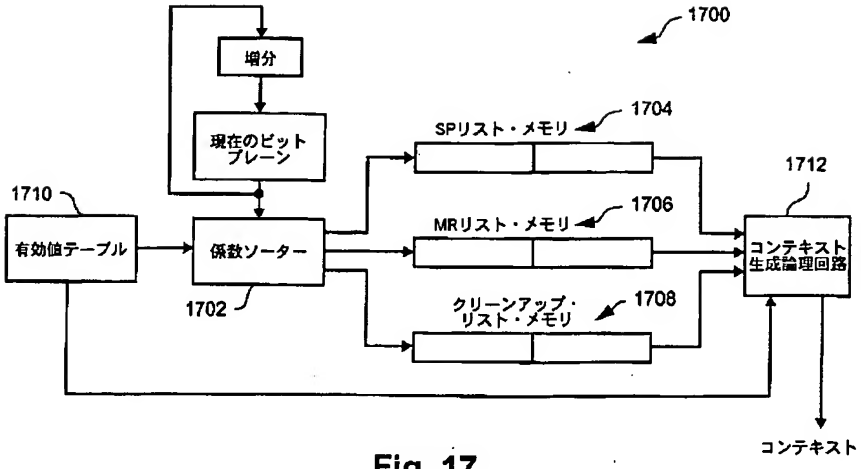


Fig. 17

【図１４】

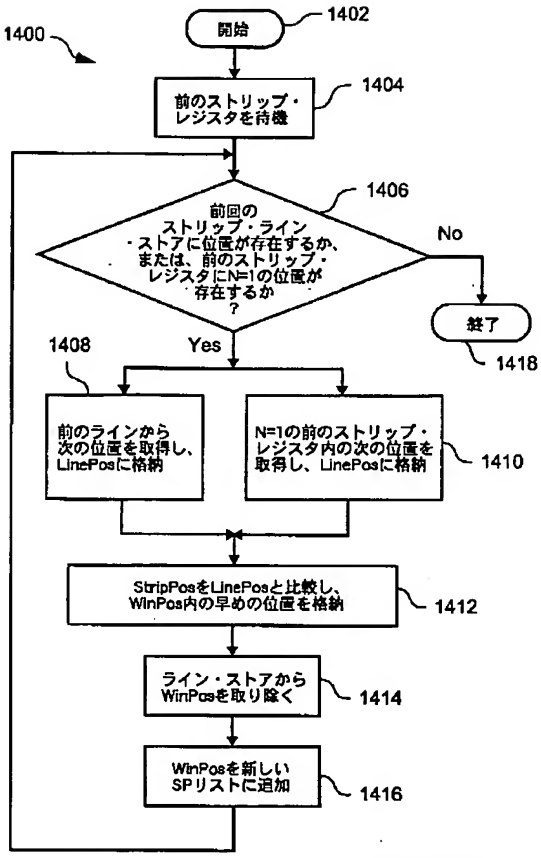


Fig. 14

【図１６】

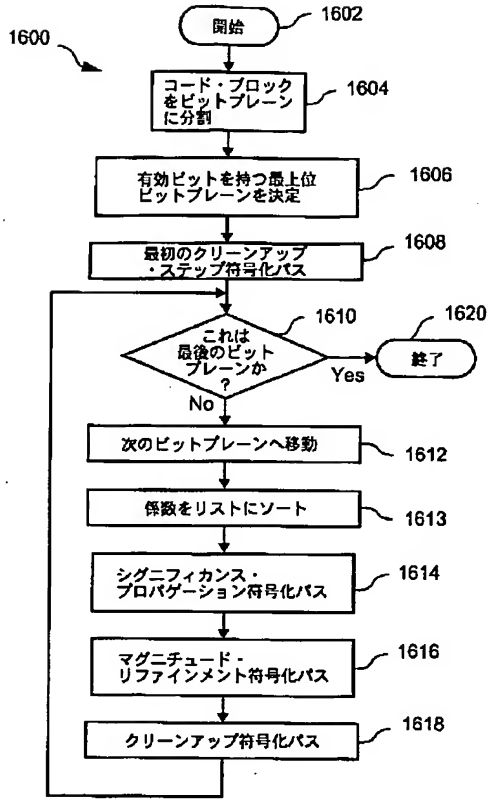


Fig. 16

【図１８】

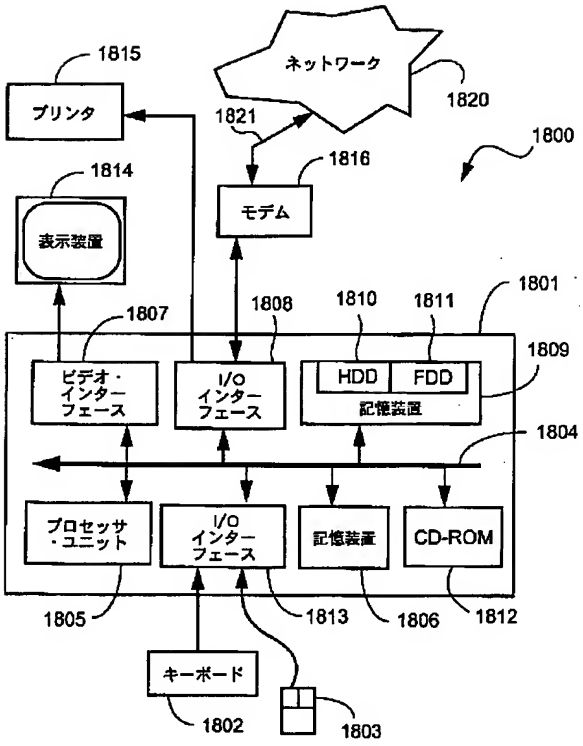


Fig. 18

フロントページの続き

Ｆターム(参考) 5C059 MA00 MA24 MA35 MD07 ME01
UA02 UA05 UA15 UA34 UA39
5J064 BA09 BA16 BC01 BC02 BC04
BD04

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-159009

(43)Date of publication of application : 31.05.2002

---

(51)Int.Cl. H04N 7/30
H03M 7/30
H03M 7/40

---

(21)Application number : 2001-261489 (71)Applicant : CANON INC

(22)Date of filing : 30.08.2001 (72)Inventor : DOMINICK IPPU

---

(30)Priority
Priority number : 2000 PQ9824
Priority date : 01.09.2000
Priority country : AU

---

(54) METHOD AND DEVICE FOR ENTROPY CODING AND DECONDING

(57)Abstract:
PROBLEM TO BE SOLVED: To provide an entropy coding method by which symbols indicating code blocks having the transformation coefficients of a digital image can be entropy coded.
SOLUTION: In this entropy coding method, a significance propagation path 314, a

magnitude refinement path 316, and a cleanup path 318 are used for entropy coding the symbols. In the method, a first coefficient list of the positions in code blocks having a symbol that becomes the object of entropy coding in the significance propagation path 314 of the present bit plane, and a second list of the positions of the coefficients having a symbol that becomes the object of the entropy coding in the magnitude refinement path 316 of the present bit plane, are generated. In addition, a third list of the positions of the coefficients which have a symbol that becomes the object of entropy coding in the cleanup path 318 of the present bit plane and are the factors in the code blocks is also generated.

LEGAL STATUS [Date of request for examination]
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

CLAIMS

[Claim(s)]

[Claim 1] It is the approach of carrying out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image. About each bit plane from the 1st bit plane to the predetermined lowest bit plane It is the multiplier which has the significant condition of zero in a current bit plane, and is a symbol belonging to a multiplier with the near multiplier which has the significant condition of one in a current bit plane. SIG which performs entropy code modulation of all the symbols in a current bit plane -- with a NIFIKANSU propagation (significance propagation) step The magnitude refinement MENTO step which performs entropy code modulation of all the symbols in the current bit

plane which is a symbol belonging to the multiplier which has the significant condition of one in the last bit plane, said SIG of a current bit plane -- among a NIFIKANSU propagation step or said magnitude refinement MENTO step The clean-up step which performs entropy code modulation of all the symbols in the current bit plane which is the symbol by which entropy code modulation was not carried out before it is performed. furthermore, said SIG of the present bit plane -- among a NIFIKANSU propagation step The list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 1st said SIG of a current bit plane -- with the step which precedes with a NIFIKANSU propagation step and is generated The inside of said magnitude refinement MENTO step of a current bit plane, The list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 2nd The step which precedes with said magnitude refinement MENTO step of a current bit plane, and is generated, It is a multiplier with the symbol set as the object of entropy code modulation among said clean-up step of a current bit plane. The approach characterized by having the step which precedes the list [of the location of said multiplier in said Cord Brock] of the 3rd with said clean-up step of a current bit

plane, and generates it.

[Claim 2] The approach that said step which generates said 1st and 2nd lists is characterized by being carried out among said clean-up step of the last bit plane in an approach according to claim 1.

[Claim 3] said step which generates said 3rd list in an approach according to claim 1 -- said SIG of a current bit plane -- the approach characterized by being carried out into a NIFIKANSU propagation step.

[Claim 4] an approach according to claim 1 -- setting -- said SIG of a current bit plane -- a NIFIKANSU propagation step The substep which is a symbol corresponding to said location in the 1st [said] list generated by said clean-up step of the last bit plane and which performs entropy code modulation of the symbol in a current bit plane, the symbol of the arbitration with the non-significant condition of zero in a current bit plane -- it is -- said SIG of a current bit plane -- in a NIFIKANSU propagation step as near which changes to the significant condition of one said SIG of a current bit plane -- among a NIFIKANSU propagation step The substep which performs entropy code modulation of the symbol of the arbitration which follows said near in order of a predetermined scan which is the symbol of the arbitration which has a multiplier

with the symbol by which entropy code modulation was carried out beforehand, The location of the multiplier with the non-significant condition of zero of a code book, said SIG of a current bit plane with the significant condition of 1 -- with the substep which generates the list [of the location of the multiplier which adjoins said multiplier which has the symbol by which entropy code modulation is carried out at a NIFIKANSU propagation step] of the 4th said SIG of a current bit plane -- with the substep which tags said location of said list of the 1st of said multipliers which change to the significant condition of one among a NIFIKANSU propagation step said SIG of a current bit plane -- with the substep which tags said location of said list of the 4th of said multipliers which change to the significant condition of one among a NIFIKANSU propagation step The approach characterized by having the substep which generates said list [of the location of the multiplier in said Cord Brock] of the 3rd.

[Claim 5] In an approach according to claim 4 said clean-up step It is the symbol generated by the NIFIKANSU propagation step. said SIG of a current bit plane -- The substep which is a symbol corresponding to said location in said 3rd list and which performs entropy code modulation of the symbol in a current bit plane, It is the multiplier which has the symbol by which entropy code modulation was

carried out among said clean-up step of a current bit plane. The substep which generates the list [of the location of said multiplier in said Cord Brock who is the multiplier which changes to the significant condition of one among said clean-up of a current bit plane] of the 5th, It is the multiplier which has the symbol by which entropy code modulation was carried out among said clean-up step of a current bit plane. Of said clean-up of a current bit plane The approach characterized by having the substep which generates the list [of the location of said multiplier in said Cord Brock who is a multiplier with the significant condition of zero] of the 6th.

[Claim 6] In an approach according to claim 5 said clean-up step The substep sorted on the list [of the location which was able to attach said tag for said 1st list] of the 7th, and the list [of the location which was not able to attach a tag] of the 8th, The approach characterized by having the substep sorted on the list [of the location which was able to attach said tag for said 4th list] of the 9th, and the list [of the location which was not able to attach a tag] of the 10th.

[Claim 7] In an approach according to claim 6, said clean-up step for which a current bit plane is used Said location of said 2nd, 5th, 7th, and 9th list is compared and merged. The substep which generates said list [of the location

for the next bit planes after a current bit plane] of the 2nd, The substep which said location of said 6th, 8th, and 10th list is compared and merged, and generates said list [of the location for the next bit planes after a current bit plane] of the 1st, The approach characterized by having the substep which removes the location of the arbitration of the multiplier in said Cord Brock which is a multiplier with the significant condition of 1 from said 1st generated list.

[Claim 8] It is the approach characterized by being the next bit plane below said Cord Brock's top bit plane in which said 1st bit plane has a significant multiplier in an approach according to claim 1.

[Claim 9] The approach characterized by having the clean-up step of the carrying-out-entropy code modulation of all symbols in said top bit plane of said Cord Brock beginning in an approach according to claim 8.

[Claim 10] In an approach according to claim 9 said first clean-up step said SIG to said 1st bit plane -- at a NIFIKANSU propagation step The substep which generates said list [of the location of the multiplier in said Cord Brock which is a multiplier with the symbol set as the object of entropy code modulation] of the 1st, At said magnitude refinement MENTO step to said 1st bit plane The approach characterized by having the substep which generates said list [of the

location of the multiplier in said Cord Brock which is a multiplier with the symbol set as the object of entropy code modulation] of the 2nd.

[Claim 11] In an approach according to claim 9 said first clean-up step About said each encoded symbol during said first clean-up step coding pass The substep which is a substep which determines said significant condition of said multiplier to which said symbol by which entropy code modulation was carried out belongs, and sets the 1st flag when said significant condition is 1, The substep which performs the substep which determines said near multiplier of said multiplier which has said predetermined significant condition of 1, and sets the 2nd flag to 1 about said each near multiplier, The substep which sets said 1st flag to zero on the other hand when said predetermined significant condition is zero, The inside of said magnitude refinement MENTO step for which said 1st bit plane is used, It is the substep which generates said list [of the location of the multiplier in said Cord Brock which is a multiplier with the symbol set as the object of entropy code modulation] of the 2nd. The substep constituted so that said location of said multiplier may be added to said 2nd list when said 1st flag relevant to said location is 1, said SIG where said 1st bit plane is used -- among a NIFIKANSU propagation step It is the substep which generates said list [of the

location of said multiplier in said Cord Brock who is a multiplier with said symbol set as the object of entropy code modulation] of the 1st. The approach characterized by having the substep constituted so that said location of said multiplier may be added to said 1st list when said 2nd flag relevant to said location is 1 and said 1st flag relevant to said location is zero.

[Claim 12] The approach characterized by said entropy-code-modulation method being able to perform the entropy code modulation of said symbol, or a decryption in an approach according to claim 1.

[Claim 13] said step which said entropy-code-modulation method is an approach of performing entropy code modulation of said symbol, in an approach according to claim 1, and generates said 1st [the], the 2nd, and the 3rd list for current bit planes -- said SIG of a current bit plane -- the approach characterized by preceding with a NIFIKANSU propagation step and being generated.

[Claim 14] Said approach of generating said list of the 2nd of current bit planes in an approach according to claim 13 is an approach characterized by having the substep which adds said location of the multiplier in said Cord Brock to said 2nd list with the significant condition more than the current bit plane number N.

[Claim 15] The approach characterized by to have the substep which adds said

location of the multiplier of the arbitration in said Cord Brock which is an approach according to claim 13 or 14, is a multiplier with the significant condition of under the current bit plane number N in said approach of generating said list of the 1st of current bit planes, and is the multiplier preceded with the near multiplier multiplier which has the significant condition more than N in order of a predetermined scan to said 1st list.

[Claim 16] In the equipment which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image The bit plane splitter which divides said Cord Brock into the bit plane which consists of a symbol from the 1st bit plane to the predetermined lowest bit plane, SIG -- with NIFIKANSU propagation pass and magnitude refinement MENTO pass The entropy encoder which encodes said symbol during clean-up pass, SIG of a current bit plane -- during NIFIKANSU propagation pass The list [of the location of said multiplier in Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 1st Precede with NIFIKANSU propagation pass and it generates. said SIG of a current bit plane -- During the magnitude refinement MENTO pass of a current bit plane The list [of the location of said multiplier in Cord Brock who is a multiplier with the symbol set as

the object of entropy code modulation] of the 2nd Precede with said magnitude refinement MENTO pass of a current bit plane, and it generates. It is the multiplier which has the symbol set as the object of entropy code modulation during the clean-up pass of a current bit plane. The list memory manager which precedes the list [of the location of said multiplier in Cord Brock] of the 3rd with said clean-up pass of a current bit plane, and generates it, Equipment characterized by having the storage unit which stores said 1st location list, the 2nd location list, and the 3rd location list.

[Claim 17] In equipment according to claim 16 said storage unit Said list of the 1st of bit planes n is inputted and stored by said list memory manager. Furthermore, the 1st storage unit constituted so that it may become the double buffer which is already generated and outputs said list of the 1st of stored bit planes (n+1) to coincidence, Said list of the 2nd of bit planes n is inputted by said list memory manager. The 2nd storage unit constituted so that it may become the double buffer which outputs said list of the 2nd of bit planes (n+1) which stored, were further already generated and were stored to coincidence, Said list of the 3rd of bit planes n is inputted by said list memory manager. Equipment characterized by having the 3rd storage unit constituted so that it may become

the double buffer which outputs said list of the 3rd of bit planes (n+1) which stored, were further already generated and were stored to coincidence.

[Claim 18] It is equipment characterized by generating said 1st and 2nd lists while said list memory manager is performing said entropy code modulation of said symbol during the clean-up pass of the last bit plane in equipment according to claim 16.

[Claim 19] equipment according to claim 16 -- setting -- said list memory manager -- SIG of a current bit plane -- the equipment characterized by generating said 3rd list while performing said entropy code modulation of said symbol during NIFIKANSU propagation pass.

[Claim 20] In equipment according to claim 16 said entropy encoder The entropy code modulation of said symbol in the current bit plane corresponding to said location in the 1st [said] list generated with said clean-up pass of the last bit plane said SIG of a current bit plane -- with the means performed during NIFIKANSU propagation pass the symbol of arbitration with the non-significant condition of zero -- it is -- said SIG of a current bit plane -- it is the symbol of the arbitration which has the near multiplier which has the symbol by which entropy code modulation was beforehand carried out as near which changes to the

significant condition of one during NIFIKANSU propagation pass -- the entropy code modulation of the symbol of the arbitration which follows said near in order of a predetermined scan -- said SIG of a current bit plane -- with the means performed in a current bit plane during NIFIKANSU propagation pass It ****. Said list memory manager a multiplier with the non-significant condition of zero -- it is -- said SIG of a current bit plane -- among a NIFIKANSU propagation step A means to generate the list [of the location of said multiplier in said Cord Brock who is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1 which is the symbol by which entropy code modulation is carried out] of the 4th, said SIG of a current bit plane -- during NIFIKANSU propagation pass with the means which tags the location of said list of the 1st of said multipliers which change to the significant condition of one said SIG of a current bit plane -- during NIFIKANSU propagation pass with the means which tags the location of said list of the 4th of said multipliers which change to the significant condition of one Equipment characterized by having a means to generate said list [of the location of the multiplier in said Cord Brock] of the 3rd.

[Claim 21] In equipment according to claim 20 said entropy encoder said SIG of a current bit plane -- it is the symbol generated with NIFIKANSU propagation

pass -- It has a means to perform entropy code modulation of the symbol in the current bit plane corresponding to said location in said 3rd list during the clean-up pass of a current bit plane. Said list memory manager Under said clean-up pass of a current bit plane, A means to generate the list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol by which entropy code modulation was carried out, and is the multiplier which changes to the significant condition of one among said clean-up of a current bit plane] of the 5th, It is the multiplier which has the symbol by which entropy code modulation was carried out during said clean-up pass of a current bit plane. Equipment characterized by having a means to generate the list [of the location of said multiplier in said Cord Brock who is a multiplier with the significant condition of the zero in said clean-up of a current bit plane] of the 6th.

[Claim 22] It is equipment characterized by to have a means sort said 4th list on a means sort said 1st list on the 7th location list with which said list memory manager was able to attach the tag in equipment according to claim 21, and the 8th location list which was not able to attach a tag, the 9th location list which was able to attach a tag, and the 10th location list which was not able to attach a tag.

[Claim 23] In equipment according to claim 22 said list memory manager Said

location of said 2nd, 5th, 7th, and 9th list is compared and merged. A means to generate said list [of the location for the next bit planes after a current bit plane] of the 2nd, A means for said location of said 6th, 8th, and 10th list to be compared and merged, and to generate said list [of the location for the next bit planes after a current bit plane] of the 1st, Equipment characterized by having the means which removes the location of the arbitration of the multiplier in said Cord Brock which is a multiplier with the significant condition of 1 from said 1st generated list.

[Claim 24] Equipment characterized by being the next bit plane below said Cord Brock's top bit plane in which said 1st bit plane has a significant multiplier in equipment according to claim 16.

[Claim 25] It is equipment characterized by said entropy encoder carrying out entropy code modulation of all the symbols in said top bit plane of said Cord Brock during the first clean-up step pass in equipment according to claim 24.

[Claim 26] In equipment according to claim 25 said list manager as said object for the 1st bit plane -- said SIG -- during NIFIKANSU propagation pass A means to generate said list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the

1st during said first clean-up step pass, Under said magnitude refinement MENTO pass for said 1st bit plane, Equipment characterized by having a means to generate said list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 2nd during said first clean-up step pass.

[Claim 27] In equipment according to claim 25 said list manager About said each encoded symbol during said first clean-up step coding pass It has a means to determine said significant condition of said multiplier to which said symbol by which entropy code modulation was carried out belongs. Said decision means If a means to set the 1st flag to 1 if said significant condition is 1, and said significant condition are 1 If a means to determine said near multiplier of said multiplier which has said predetermined significant condition of 1, and to set the 2nd flag to 1 about said each near multiplier, and said predetermined significant condition are zero The decision means constituted so that it may have a means to set said 1st flag to zero, Under said magnitude refinement MENTO pass for said 1st bit plane, It has a means to generate said list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 2nd during said first clean-up step coding

pass. The means constituted so that said location of said multiplier may be added to said 2nd list when said 1st flag relevant to said location is 1, said 1st bit plane -- said SIG -- during NIFIKANSU propagation pass It has a means to generate said list [of the location of said multiplier in said Cord Brock who is a multiplier with said symbol by which entropy code modulation is carried out] of the 1st. Equipment characterized by adding said location of said multiplier to said 1st list when said 2nd flag relevant to said location is 1 and said 1st flag relevant to said location is zero.

[Claim 28] Equipment characterized by said equipment of entropy code modulation being able to perform the entropy code modulation of said symbol, or decode in equipment according to claim 16.

[Claim 29] equipment according to claim 16 -- setting -- said equipment of entropy code modulation -- the entropy coding equipment of said symbol -- it is -- said 1st, 2nd, and 3rd list for bit planes current in said list memory manager -- said SIG of a current bit plane -- the equipment characterized by preceding with NIFIKANSU propagation pass and generating.

[Claim 30] It is equipment characterized by having a means to add said location of the multiplier in said Cord Brock to said 2nd list in which said list memory

manager has the significant condition more than the current bit plane number N in equipment according to claim 29.

[Claim 31] It is equipment characterized by having a means to add said location of the multiplier of the arbitration in said Cord Brock which is the multiplier which said list memory manager has the significant condition of under the current bit plane number N, and has the significant condition more than N in order of a predetermined scan in equipment according to claim 29 to said 1st list to be preceded with a multiplier soon.

[Claim 32] It is the computer program product which has the computer program which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image. A code for said computer program to divide said Cord Brock into the bit plane which consists of a symbol from the 1st bit plane to the predetermined lowest bit plane, SIG -- with NIFIKANSU propagation pass and magnitude refinement MENTO pass The code for carrying out entropy code modulation of said symbol during clean-up pass, SIG of a current bit plane -- during NIFIKANSU propagation pass The list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 1st said SIG of a

current bit plane -- with the code for preceding with NIFIKANSU propagation pass and generating Under the magnitude refinement MENTO pass of a current bit plane, The list [of the location of said multiplier in said Cord Brock who is a multiplier with the symbol set as the object of entropy code modulation] of the 2nd The code for preceding with said magnitude refinement MENTO pass of a current bit plane, and generating, Have the symbol set as the object of entropy code modulation during the clean-up pass of a current bit plane. The code for preceding the list [of the location of said multiplier in said Cord Brock] of the 3rd with said clean-up pass of a current bit plane, and generating it, The computer program product characterized by having a code for storing said 1st location list, the 2nd location list, and the 3rd location list.

[Claim 33] It is the entropy-code-modulation approach of the symbol showing Brock who has a multiplier. About each bit plane from the 1st bit plane to the predetermined minimum bit plane The step which generates at least one list of the location of said multiplier in Brock who has the symbol by which entropy code modulation is carried out among said Brock's current bit plane, The approach characterized by performing the step which carries out entropy code modulation of said symbol of the multiplier in said Brock's current bit plane using

said at least one list.

[Claim 34] The equipment characterized by to have the generation unit which generates at least one list of the location of said multiplier in said Brock who is equipment which carries out entropy code modulation of the symbol showing Brock who has a multiplier, and has the symbol by which entropy code modulation is carried out among said Brock's current bit plane, and the entropy encoder which carries out entropy code modulation of said symbol of the multiplier in said Brock's current bit plane using said at least one list.

[Claim 35] The computer program product characterized by to have a code for generating at least one list of the location of said multiplier in said Brock who is the computer program product which has the computer program which carries out entropy code modulation of the symbol showing Brock who has a multiplier, and has the symbol by which entropy code modulation is carried out among said Brock's current bit plane, and a code for carrying out entropy code modulation of said symbol of the multiplier in said Brock's current bit plane using said at least one list.

[Claim 36] It is the approach of carrying out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image.

About each bit plane from the 1st bit plane to the predetermined lowest bit plane the 1st [of the multiplier which has said symbol set as the object of entropy code modulation in a current bit plane, respectively], 2nd, and 3rd location lists -- following -- SIG -- with NIFIKANSU propagation pass Magnitude refinement MENTO pass and the step which encodes the symbol of said transform coefficient said Cord Brock's current bit plane during clean-up pass, The list [of the location for the next bit planes after a current bit plane] of the 1st, The list [of the location for the next bit planes after a current bit plane] of the 2nd, It has the step which generates the list [of the location for current bit planes] of the 3rd: Said step to generate a multiplier with the non-significant condition of zero -- it is -- said SIG of a current bit plane -- it is the symbol by which entropy code modulation is carried out during NIFIKANSU propagation pass -- It is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1. the list [of the location of said multiplier in said Cord Brock] of the 4th -- said SIG of a current bit plane -- with the substep generated during NIFIKANSU propagation pass said SIG of a current bit plane -- it is the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass -- the location of the list [of the location of the multiplier in a current bit plane] of

the 1st -- said SIG of a current bit plane -- with the substep which attaches a tag with NIFIKANSU propagation pass said SIG of a current bit plane -- on the list [of said location of the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass] of the 4th said SIG of a current bit plane -- with the substep which attaches a tag during NIFIKANSU propagation pass It is the substep which generates said list [of the location for current bit planes] of the 3rd during NIFIKANSU propagation pass. SIG of a current bit plane -- Said 3rd list The location in said 2nd list of the location for current bit planes, said SIG -- with the location of a multiplier with the symbol by which entropy code modulation was carried out during NIFIKANSU propagation pass The substep constituted so that all ****** Cord Brock's location lists may be included, It is a multiplier with the symbol by which entropy code modulation was carried out during said clean-up pass of a current bit plane. The substep which generates the list [of the location of said multiplier in Cord Brock who changes to the significant condition of one during the clean-up pass of a current bit plane, and who is a multiplier] of the 5th during the clean-up pass of a current bit plane, It has the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. It is the multiplier which has the significant

condition of zero during the clean-up pass of a current bit plane. The substep which generates the list [of the location of said multiplier in Cord Brock] of the 6th during said clean-up pass of a current bit plane, The 1st location list which attached said tag during said clean-up pass of a current bit plane, The 1st location list which did not attach a tag is sorted about a current bit plane. The substep changed into the 7th location list and the 8th location list, respectively, The 4th location list which attached said tag during said clean-up pass of a current bit plane, The 4th location list which did not attach a tag is sorted about a current bit plane. The substep changed into the 9th location list and the 10th location list, respectively, Said location on the said 2nd, 5th, 7th, and 9th lists is merged [be / it / under / clean-up pass / of a current bit plane / comparison]. The substep which generates said list [of the location for the next bit planes after a current bit plane] of the 2nd, Said location on the said 6th, 8th, and 10th lists is merged [be / it / under / clean-up pass / of a current bit plane / comparison]. The substep which generates said list [of the location for the next bit planes after a current bit plane] of the 1st, The approach characterized by having the substep which is a multiplier with the significant condition of 1, and which removes the location of the multiplier of the arbitration in Cord Brock from said 1st generated

list of [for the next bit plane] during the clean-up pass of a current bit plane.

[Claim 37] In the equipment which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image The 1st which has the symbol by which entropy code modulation was carried out in the current bit plane, respectively, the 2nd and the location list of the 3rd multiplier -- following -- SIG -- with NIFIKANSU propagation pass Magnitude refinement MENTO pass and the entropy encoder which encodes said symbol of said transform coefficient Cord Brock's current bit plane during clean-up pass, Said list [of the location for the next bit planes after a current bit plane] of the 1st, The list [of the location for the next bit planes after a current bit plane] of the 2nd, It has the list memory manager which generates the list [of the location for current bit planes] of the 3rd. Said list memory manager a multiplier with the non-significant condition of zero -- it is -- said SIG of a current bit plane -- it is the symbol by which entropy code modulation is carried out during NIFIKANSU propagation pass -- the list [of the location of said multiplier in said Cord Brock who is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1] of the 4th -- said SIG of a current bit plane -- with a means to generate during NIFIKANSU propagation pass said SIG of a current bit plane --

it is the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass -- the location of the list of the 1st of multipliers of the location in a current bit plane -- said SIG of a current bit plane -- with the means which attaches a tag during NIFIKANSU propagation pass said SIG of a current bit plane -- on the list [of said location of the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass] of the 4th said SIG of a current bit plane -- with the means which attaches a tág during NIFIKANSU propagation pass It is a means to generate said list [of the location for current bit planes] of the 3rd during NIFIKANSU propagation pass. said SIG of a current bit plane -- Said 3rd list The location in said 2nd list of the location for current bit planes, said SIG -- with the location of a multiplier with the symbol by which entropy code modulation was carried out during NIFIKANSU propagation pass The means constituted so that all ****** Cord Brock's location lists may be included, It is a multiplier with the symbol by which entropy code modulation was carried out during said clean-up pass of a current bit plane. It is the multiplier which changes to the significant condition of one during said clean-up pass of a current bit plane. A means to generate the list [of the location of said multiplier in said Cord Brock] of the 5th during said clean-up pass of a current bit plane, It is

a multiplier with the symbol by which entropy code modulation was carried out during said clean-up pass of a current bit plane. A means to generate the list [of the location of said multiplier in said Cord Brock who is the multiplier which has the significant condition of zero during said clean-up pass of a current bit plane] of the 6th during said clean-up pass of a current bit plane, The 1st location list which attached said tag during said clean-up pass of a current bit plane, The 1st location list which did not attach a tag is sorted about a current bit plane. The means changed into the 7th location list and the 8th location list, respectively, The 4th location list which attached said tag during said clean-up pass of a current bit plane, The 4th location list which did not attach a tag is sorted about a current bit plane. The means changed into the 9th location list and the 10th location list, respectively, Said location on the said 2nd, 5th, 7th, and 9th lists is merged [be / it / under / said clean-up pass / of a current bit plane / comparison]. A means to generate said list [of the location for the next bit planes after a current bit plane] of the 2nd, A means for said location on the said 6th, 8th, and 10th lists to be merged [be / it / under / said clean-up pass / of a current bit plane / comparison], and to generate said list [of the location for the next bit planes after a current bit plane] of the 1st, Equipment characterized by having the

means which removes the location of the multiplier of the arbitration in said Cord Brock which is a multiplier with the significant condition of 1 from said 1st generated list of [for the next bit plane] during said clean-up pass of a current bit plane.

[Claim 38] It is the computer program product which has the computer program which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image. The 1st of a multiplier with the symbol to which entropy code modulation of said computer program was carried out in the current bit plane, respectively, the 2nd and the 3rd location list -- following -- SIG -- with NIFIKANSU propagation pass Magnitude refinement MENTO pass and the code for carrying out entropy code modulation of said symbol of said transform coefficient Cord Brock's current bit plane during clean-up pass, Said list [of the location for the next bit planes after a current bit plane] of the 1st, The list [of the location for the next bit planes after a current bit plane] of the 2nd, the code for generating the list [of the location for current bit planes] of the 3rd -- having -- the business for [said] generating -- a code a multiplier with the non-significant condition of zero -- it is -- said SIG of a current bit plane -- it is the symbol by which entropy code modulation is carried out

during NIFIKANSU propagation pass -- It is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1. the list [of the location of said multiplier in said Cord Brock] of the 4th -- said SIG of a current bit plane -- with the code generated during NIFIKANSU propagation pass said SIG of a current bit plane -- it is the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass -- the location of the list [of the location of the multiplier in a current bit plane] of the 1st -- said SIG of a current bit plane -- with the code which attaches a tag during NIFIKANSU propagation pass As opposed to the location of the list [of said location of the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass] of the 4th said SIG of a current bit plane -- said SIG of a current bit plane -- with the code which attaches a tag during NIFIKANSU propagation pass It is the code which generates said list [of the location for current bit planes] of the 3rd during NIFIKANSU propagation pass. said SIG of a current bit plane -- Said 3rd list The location in said 2nd list of the location for current bit planes, said SIG -- with the location of a multiplier with the symbol by which entropy code modulation was carried out during NIFIKANSU propagation pass The code constituted so that all ****** Cord Brock's location lists may be included, It has

the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. Furthermore, the code which generates the list [of the location of said multiplier in Cord Brock who changes to the significant condition of one] of the 5th during said clean-up pass of the present bit plane during the clean-up pass of the present bit plane, It has the symbol by which entropy code modulation was carried out during said clean-up pass of a current bit plane. The code which generates the list [of the location of said multiplier in Cord Brock who is the multiplier which has the significant condition of zero during said clean-up pass of a current bit plane] of the 6th during said clean-up pass of a current bit plane, The 1st location list which attached said tag during said clean-up pass of a current bit plane, The 1st location list which did not attach a tag is sorted about a current bit plane. The code changed into the 7th location list and the 8th location list, respectively, The 4th location list which attached said tag during said clean-up pass of a current bit plane, The 4th location list which did not attach a tag is sorted about a current bit plane. The code changed into the 9th location list and the 10th location list, respectively, Said location on the said 2nd, 5th, 7th, and 9th lists is merged [be / it / under / said clean-up pass / of a current bit plane / comparison]. The code which

generates said list [of the location for the next bit planes after a current bit plane] of the 2nd, The code which said location on the said 6th, 8th, and 10th lists is merged [be / it / under / said clean-up pass / of a current bit plane / comparison], and generates said list [of the location for the next bit planes after a current bit plane] of the 1st, The code which removes the location of the multiplier of the arbitration in said Cord Brock which is a multiplier with the significant condition of 1 from said 1st generated list of [for the next bit plane] during said clean-up pass of a current bit plane, The computer program product characterized by ****(ing).

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention] Generally especially this invention relates to entropy code modulation, the decryption approach, and its equipment about a data compression and defrosting. This invention relates to the computer program product which has a computer program for performing entropy code modulation again.

[0002]

[Description of the Prior Art] The request for proposal to JPEG-2000 new specification was advanced recently, and the specification draft (on these specifications, henceforth referred to as JPEG2000) of a title called the "information technology-JPEG2000 image coding system-JPEG2000 committee

draft version 1.0 (December 9, 1999) was announced.

[0003] In JPEG2000, the whole image is divided into one or more image tile components, and, subsequently performing each two-dimensional dispersion wavelet transform of this image tile component is proposed. Subsequently, grouping of the transform coefficient of each image tile component is carried out to a subband. Subsequently, before entropy code modulation of each code block is carried out, this subband is divided and is changed into rectangular Cord Brock. Each code Brock's transform coefficient is preceded with entropy code modulation, and is expressed with sign magnitude. Entropy code modulation consists of two parts, the context generation section and the algebraic-sign-ized section. The algebraic-sign-ized section is considering the bit symbol and the context of a bit symbol of the multiplier set as the object of coding as the input. The context of the bit symbol of a certain multiplier is based on the 'significant (significance)' condition of the envelopment multiplier of eight pieces in Cord Brock's same bit plane. This bit symbol is encoded by the algebraic-sign-ized section. Although the 'significant' condition of a multiplier is the binary-value variable Si [m, n] and this binary-value variable is initialized by 0, this variable changes to 1 immediately after encoding the 1st non zero bit value of a multiplier.

Drawing 1 The near significant condition Si of the envelopment multiplier of eight pieces of a multiplier "X [m, n]" [m-1, n-1] Si [m-1, n], Si [m-1, n+1], Si [m, n-1], Si [m, n+1], Si [m+1, n-1], Si [m+1, n], and Si [m+1, n+1] are shown. However, m and n are Cord Brock's line numbers and row numbers, respectively, and Si[] is significant condition just before the bit symbol i of a multiplier X [m, n] is encoded. Such near significant condition may be called about [3x] 3 significant condition of a multiplier X [m, n].

[0004] The algebraic-sign-ized section [m, n] encodes first all the bit symbols of the one top bit plane of Cord Brock, subsequently to Cord Brock's degree encodes **, such as all bit symbols of a low bit plane, and, finally encodes all the bit symbols of the lowest bit plane. Within the limits of each one bit plane of Cord Brock, the algebraic-sign-ized section encodes the bit symbol of a multiplier with three pass of predetermined sequence.

[0005] the 1st pass of a bit plane -- SIG -- it is called NIFIKANSU propagation pass (SP pass), the 2nd pass of a bit plane is called magnitude refinement MENTO pass (MR pass), and the pass of the 3rd last of a bit plane is called clean-up pass (N pass). Although the bit symbol for coding has the significant condition of 1 in SP pass, when this bit symbol itself has the near bit symbol with

the significant condition of 0, the bit symbol of a bit plane is encoded. If the multiplier of this bit symbol is the last coding bit plane and is already significant in MR pass when the bit symbol of a bit plane is not encoded yet, this bit symbol will be encoded. The remaining bit symbols of the bit plane which was not beforehand encoded in N pass are encoded at this time.

[0006] This context is transmitted to the algebraic-sign-ized section with the bit for coding, and the encoded symbol is outputted to a bit stream. If the value of this bit symbol for coding is 1 and that significant condition is zero, a bit symbol is once encoded, and significant condition will be set to 1 when the next bit symbol for coding is a sign bit for multipliers. When not fulfilling the above-mentioned conditions, significant condition is still zero (0). When taking the context of a continuous multiplier and pass into consideration, the most current significant condition of this multiplier is used.

[0007] The algebraic-sign-ized section encodes the bit symbol of the bit plane in three pass (SP, MR, N) in the same predetermined sequence. The algebraic-sign-ized section goes to the top bit plane which has a non zero bit in inside first, skips SP and MR pass, and begins from N pass. Subsequently, the algebraic-sign-ized section goes to a bit plane low next, are three pass (SP, MR,

N), and encodes a bit symbol in the sequence. Subsequently, the algebraic-sign-ized section goes to a bit plane low next, encodes **, such as a bit symbol, with the pass (SP, MR, N) of same order, and, finally encodes the lowest bit plane.

[0008] Furthermore, each bit plane of Cord Brock is scanned in special sequence. It begins from topmost part left-hand side, and the first four bit symbols of a train are scanned. Subsequently, the first four bit symbols of the 2nd train are scanned until Cord Brock's width of face will be covered. Subsequently, it is scanned with **, such as the 2nd four bit symbol of the 1st train. About the remaining lines of arbitration, the same scan is continued within lowest Cord Brock of a subband. Drawing 2 shows an example of the Cord Brock scanning pattern about Cord Brock with the transform coefficient of 64 pieces which consists of 8x8 Brock. A scan is performed by the strip (long and slender piece) which four lines follow so that it may understand in this example. Cord Brock is not limited to 8x8 Brock, and it thinks of still bigger Cord Brock, such as Cord Brock of 64x64. In the case of the latter, 16 continuous strips of four lines arise. In order to simplify explanation, on these specifications, this order of a scan will be called the order of JPEG2000 scan.

[0009] The entropy decryption described by JPEG2000 becomes the relation between entropy code modulation and a mirror image. For example, a decoder decodes a symbol by the case of coding, and same order. An entropy decryption consists of two sections, the context generation section and the arithmetic decode section, again. The arithmetic decryption section receives as an input the context of the symbol set as the object of decode, and its symbol set as the object of decode. This symbol is decoded by the arithmetic decryption section based on the 'significant' condition of the envelopment multiplier of eight pieces in the bit plane where Cord Brock of the context of the scan sequence symbol set as the object of decode is the same. Although the 'significant' condition of a multiplier is binary-value variable S [m, n] and this binary-value variable is initialized by 0, if the 1st non zero bit plane value of a multiplier is decoded, it will change to 1. Thus, the significant condition of the multiplier in a decryption phase becomes the significant condition and mirror image relation of a multiplier in a coding phase (refer to drawing 2).

[0010] In JPEG2000, algebraic-sign-izing and a decryption are performed for every bit plane from the top bit plane to the lowest bit plane. Introduction, and coding/decryption skip all the bit planes that contain only zero in it, and it starts

the operation to the 1st bit plane which has a non zero element in it. once it arrives at the 1st bit plane with which coding/decryption has a non zero element in it -- SIG -- a codec acts to each continuation bit plane with NIFIKANSU propagation pass, magnitude refinement MENTO pass, and three pass of clean-up pass.

[0011] It has the codec scan covering the whole bit plane, and the entropy codec which realizes JPEG2000 which defines arrangement of a location with a certain description is proposed. for example, SIG -- with NIFIKANSU propagation pass, a codec looks for the location which has at least one significant multiplier near 3x3 of the location, although the location itself is not significant. With magnitude refinement MENTO pass, a codec already looks for a significant location. With clean-up pass, a codec looks for the location which is not encoded / decrypted in front. Therefore, in this proposed codec, three scans are performed about all bit planes. This connotes that the maximum throughput is one symbol per 3 cycles.

[0012] As main troubles of the proposed codec, about each pass, a codec scans all locations and the point that it is necessary to judge whether coding/decryption is performed by current pass is mentioned so that the above may show. This means that scan a location without the need for coding and a cycle is wasted.

[0013]

[Problem(s) to be Solved by the Invention] Solution or improving at least are the purpose of this invention substantially about one or more troubles of this configuration and/or a proposal.

[0014]

[Means for Solving the Problem] According to the 1st mode of this invention, the approach of carrying out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image is offered. The following steps [bit plane / to the lowest bit plane predetermined / the 1st bit plane to / in this approach / each] : It is the multiplier which has the significant condition of zero in a current bit plane. SIG which performs entropy code modulation of all the symbols in the current bit plane which is a symbol belonging to a multiplier with the near multiplier which has the significant condition of one in a current bit plane -- with a NIFIKANSU propagation step ; The magnitude refinement MENTO step which performs entropy code modulation of all the symbols in the current bit plane which is a symbol belonging to the multiplier which has the significant condition of one in the last bit plane, ; said SIG of a current bit plane -- among a NIFIKANSU propagation step or a magnitude

refinement MENTO step In the approach of performing the clean-up step which performs entropy code modulation of all the symbols in the current bit plane which is the symbol by which entropy code modulation was not carried out before it SIG of a current bit plane -- it is a multiplier with the symbol set as the object of the entropy code modulation in a NIFIKANSU propagation step -- the list [of the location of said multiplier in said Cord Brock] of the 1st -- said SIG of a current bit plane -- with the step which precedes with a NIFIKANSU propagation step and is generated ; It is a multiplier with the symbol set as the object of the entropy code modulation in a magnitude refinement MENTO step of a current bit plane. The step which precedes the list [of the location of said multiplier in said Cord Brock] of the 2nd with the magnitude refinement MENTO step of a current bit plane, and generates it, ; It is a multiplier with the symbol set as the object of the entropy code modulation in said clean-up step of a current bit plane. It has further the step which precedes the list [of the location of said multiplier in said Cord Brock] of the 3rd with the clean-up step of a current bit plane, and generates it.

[0015] According to another mode of this invention, the equipment which carries out entropy code modulation of the symbol showing Cord Brock who has the

transform coefficient of a digital image is offered. the bit plane splitter with which this equipment divides said Cord Brock into the bit plane which consists of a symbol from the 1st bit plane to the predetermined lowest bit plane, and; SIG -- with NIFIKANSU propagation pass SIG of magnitude refinement MENTO pass, the entropy encoder which encodes said symbol during clean-up pass, and a; current bit plane -- it is a multiplier with the symbol set as the object of entropy code modulation during NIFIKANSU propagation pass -- Precede with NIFIKANSU propagation pass and it generates. the list [of the location of said multiplier in Cord Brock] of the 1st -- SIG of a current bit plane -- It is a multiplier with the symbol set as the object of entropy code modulation during the magnitude refinement MENTO pass of a current bit plane. Precede the list [of the location of said multiplier in Cord Brock] of the 2nd with the magnitude refinement MENTO pass of a current bit plane, and it is generated. It is a multiplier with the symbol set as the object of entropy code modulation during the clean-up pass of a current bit plane. The list memory manager which precedes the list [of the location of said multiplier in Cord Brock] of the 3rd with the clean-up pass of a current bit plane, and generates it; it has the storage which stores said 1st list, the 2nd list, and the 3rd location list.

[0016] According to another mode of this invention, the computer program product which has the computer program which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image is offered. A code for this computer program to divide said Cord Brock into the bit plane which consists of a symbol from the 1st bit plane to the predetermined lowest bit plane, SIG -- with NIFIKANSU propagation pass and magnitude refinement MENTO pass The code for carrying out entropy code modulation of said symbol during clean-up pass, SIG of a current bit plane -- it is a multiplier with the symbol set as the object of entropy code modulation during NIFIKANSU propagation pass -- the list [of the location of said multiplier in Cord Brock] of the 1st -- SIG of a current bit plane -- with the code for preceding with NIFIKANSU propagation pass and generating It is a multiplier with the symbol set as the object of entropy code modulation during the magnitude refinement MENTO pass of a current bit plane. The code for preceding the list [of the location of said multiplier in Cord Brock] of the 2nd with the magnitude refinement MENTO pass of a current bit plane, and generating it, It is a multiplier with the symbol set as the object of entropy code modulation during the clean-up pass of a current bit plane. It has a code for storing the code for preceding the

list [of the location of said multiplier in Cord Brock] of the 3rd with the clean-up pass of a current bit plane, and generating it, said 1st location list, the 2nd location list, and the 3rd location list.

[0017] According to another mode of this invention, the approach of the entropy coding symbol showing Brock who has a multiplier is offered. The following steps [bit plane / to the lowest bit plane predetermined / the 1st bit plane to / in this approach / each]: Perform the step which generates at least one list of the location of said multiplier in Brock who is the multiplier which has Brock's current symbol by which entropy code modulation in a bit plane is carried out, and the step which carries out entropy code modulation of said symbol of the multiplier in the Brock's current bit plane using said at least one list.

[0018] According to another mode of this invention, the equipment which carries out entropy code modulation of the symbol showing Brock who has a multiplier is offered. This equipment has the generation equipment which generates at least one list of the location of said multiplier in Brock who is a multiplier with Brock's current symbol by which entropy code modulation in a bit plane is carried out, and the entropy encoder which carries out entropy code modulation of said symbol of the multiplier in the Brock's current bit plane using said at least one list.

[0019] According to another mode of this invention, the computer program product which has the computer program which carries out entropy code modulation of the symbol showing Brock who has a multiplier is offered. This computer program has a code for generating at least one list of the location of said multiplier in Brock who is the multiplier which has Brock's current symbol by which entropy code modulation in a bit plane is carried out, and a code for carrying out entropy code modulation of said symbol of the multiplier in the Brock's current bit plane using said at least one list.

[0020] According to another mode of this invention, the approach of carrying out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image is offered. The following steps [bit plane / to the lowest bit plane predetermined / the 1st bit plane to / in this approach / each] : The 1st of the multiplier which has said symbol set as the object of entropy code modulation in a current bit plane, respectively, the 2nd and the 3rd location list -- following -- SIG -- with NIFIKANSU propagation pass Magnitude refinement MENTO pass and the step which encodes the symbol of said transform coefficient Cord Brock's current bit plane during clean-up pass, The step and ** which generate the list [of the location for the next bit planes after a

current bit plane] of the 1st, the list [of the location for the next bit planes after a current bit plane] of the 2nd, and the list [of the location for current bit planes] of the 3rd are offered. In this case, this generation step is a multiplier with the non-significant condition of zero. said SIG of a current bit plane -- it is the symbol by which entropy code modulation is carried out during NIFIKANSU propagation pass -- It is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1. the list [of the location of said multiplier in said Cord Brock] of the 4th -- SIG of a current bit plane -- with the substep generated during NIFIKANSU propagation pass SIG of a current bit plane -- it is the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass -- SIG of a bit plane current to the location of the list [of the location of the multiplier in a current bit plane] of the 1st -- with the substep which attaches a tag during NIFIKANSU propagation pass SIG of a current bit plane -- SIG of a bit plane current to the list [of said location of the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass] of the 4th -- with the substep which attaches a tag during NIFIKANSU propagation pass It is the substep which generates said list [of the location for current bit planes] of the 3rd during NIFIKANSU propagation pass. SIG of a

current bit plane -- Said 3rd list The location in said 2nd list of the location for current bit planes, SIG -- with the location of a multiplier with the symbol by which entropy code modulation was carried out during NIFIKANSU propagation pass The substep constituted so that all ****** Cord Brock's location lists may be included, It is a multiplier with the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. The substep which generates the list [of the location of said multiplier in Cord Brock who changes to the significant condition of one, and who is a multiplier] of the 5th during the clean-up pass of a current bit plane during the clean-up pass of a current bit plane, It has the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. The substep which generates the list [of the location of said multiplier in Cord Brock who is the multiplier which has the significant condition of zero during the clean-up pass of a current bit plane] of the 6th during the clean-up pass of a current bit plane, The 1st location list which attached said tag during the clean-up pass of a current bit plane, The 1st location list which did not attach a tag is sorted about a current bit plane. The substep changed into the 7th location list and the 8th location list, respectively, The 4th location list which attached said tag during the clean-up pass of a

current bit plane, The 4th location list which did not attach a tag is sorted about a current bit plane. The substep changed into the 9th location list and the 10th location list, respectively, Said location on the said 2nd, 5th, 7th, and 9th lists is merged [be / it / under / clean-up pass / of a current bit plane / comparison]. The substep which generates said list [of the location of the next bit plane after a current bit plane] of the 2nd, The substep which said location on the said 6th, 8th, and 10th lists is merged [be / it / under / clean-up pass / of a current bit plane / comparison], and generates said list [of the location of the next bit plane after a current bit plane] of the 1st, It has the substep which is a multiplier with the significant condition of 1 and which removes the location of the multiplier of the arbitration in Cord Brock from said 1st generated list of [for the next bit plane] during the clean-up pass of a current bit plane.

[0021] According to another mode of this invention, the equipment which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient of a digital image is offered. The 1st in which this equipment has the symbol by which entropy code modulation was carried out in the current bit plane, respectively, the 2nd and the location list of the 3rd multiplier -- following -- SIG -- with NIFIKANSU propagation pass Magnitude

refinement MENTO pass and the entropy encoder which encodes said symbol of said transform coefficient Cord Brock's current bit plane during clean-up pass, Said list [of the location for the next bit planes after a current bit plane] of the 1st, The list [of the location for the next bit planes after a current bit plane] of the 2nd, The list memory manager which generates the list [of the location for current bit planes] of the 3rd, It is the multiplier in which it **** and a list memory manager has the non-significant condition of zero. said SIG of a current bit plane -- it is the symbol by which entropy code modulation is carried out during NIFIKANSU propagation pass -- the list [of the location of said multiplier in said Cord Brock who is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1] of the 4th -- SIG of a current bit plane -- with a means to generate during NIFIKANSU propagation pass SIG of a current bit plane -- it is the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass -- SIG of a bit plane current to the location of the list [of the location of the multiplier in a current bit plane] of the 1st -- with the means which attaches a tag during NIFIKANSU propagation pass SIG of a current bit plane -- SIG of a bit plane current to the list [of said location of the multiplier which changes to the significant condition of one during NIFIKANSU

propagation pass] of the 4th -- with the means which attaches a tag during NIFIKANSU propagation pass It is a means to generate said list [of the location for current bit planes] of the 3rd during NIFIKANSU propagation pass. SIG of a current bit plane -- Said 3rd list The location in said 2nd list of the location for current bit planes, SIG -- with the location of a multiplier with the symbol by which entropy code modulation was carried out during NIFIKANSU propagation pass The means constituted so that all ****** Cord Brock's location lists may be included, It is a multiplier with the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. A means to generate the list [of the location of said multiplier in Cord Brock who changes to the significant condition of one and who is a multiplier] of the 5th during the clean-up pass of a current bit plane during the clean-up pass of a current bit plane, It is a multiplier with the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. A means to generate the list [of the location of said multiplier in Cord Brock who is the multiplier which has the significant condition of zero during the clean-up pass of a current bit plane] of the 6th during the clean-up pass of a current bit plane, The 1st location list which attached said tag during the clean-up pass of a current bit plane, The 1st

location list which did not attach a tag is sorted about a current bit plane. The means changed into the 7th location list and the 8th location list, respectively, The 4th location list which attached said tag during the clean-up pass of a current bit plane, The 4th location list which did not attach a tag is sorted about a current bit plane. The means changed into the 9th location list and the 10th location list, respectively, A means for said location on the said 2nd, 5th, 7th, and 9th lists to be merged [be / it / under / clean-up pass / of a current bit plane / comparison], and to generate said list [of the location for the next bit planes after a current bit plane] of the 2nd, A means for said location on the said 6th, 8th, and 10th lists to be merged [be / it / under / clean-up pass / of a current bit plane / comparison], and to generate said list [of the location for the next bit planes after a current bit plane] of the 1st, It has the means which removes the location of the multiplier of the arbitration in Cord Brock which is a multiplier with the significant condition of 1 from said 1st generated list of [for the next bit plane] during the clean-up pass of a current bit plane.

[0022] According to another mode of this invention, the computer program product which has the computer program which carries out entropy code modulation of the symbol showing Cord Brock who has the transform coefficient

of a digital image is offered. The 1st of a multiplier with the symbol to which entropy code modulation of this computer program was carried out in the current bit plane, respectively, the 2nd and the 3rd location list -- following -- SIG -- with NIFIKANSU propagation pass Magnitude refinement MENTO pass and the code for carrying out entropy code modulation of said symbol of said transform coefficient Cord Brock's current bit plane during clean-up pass, Said list [of the location for the next bit planes after a current bit plane] of the 1st, The list [of the location for the next bit planes after a current bit plane] of the 2nd, It has a code for generating the list [of the location for current bit planes] of the 3rd. In that case this code for generation a multiplier with the non-significant condition of zero -- it is -- said SIG of a current bit plane -- it is the symbol by which entropy code modulation is carried out during NIFIKANSU propagation pass -- the list [of the location of said multiplier in said Cord Brock who is said multiplier which adjoins a multiplier with a symbol with the significant condition of 1] of the 4th -- SIG of a current bit plane -- with the code generated during NIFIKANSU propagation pass SIG of a current bit plane -- it is the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass -- SIG of a bit plane current to the location of the list [of the location of the multiplier in a

current bit plane] of the 1st -- with the code which attaches a tag during NIFIKANSU propagation pass SIG of a current bit plane -- the location of the list [of said location of the multiplier which changes to the significant condition of one during NIFIKANSU propagation pass] of the 4th -- receiving -- SIG of a current bit plane -- with the code which attaches a tag during NIFIKANSU propagation pass It is the code which generates said list [of the location for current bit planes] of the 3rd during NIFIKANSU propagation pass. SIG of a current bit plane -- Said 3rd list The location in said 2nd list of the location for current bit planes, SIG -- with the location of a multiplier with the symbol by which entropy code modulation was carried out during NIFIKANSU propagation pass The code constituted so that all ****** Cord Brock's location lists may be included, It has the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. Furthermore, the code which generates the list [of the location of said multiplier in Cord Brock who changes to the significant condition of one] of the 5th during the clean-up pass of the present bit plane during the clean-up pass of the present bit plane, It has the symbol by which entropy code modulation was carried out during the clean-up pass of a current bit plane. The code which generates the list [of the location of

said multiplier in Cord Brock who is the multiplier which has the significant condition of zero during the clean-up pass of a current bit plane] of the 6th during the clean-up pass of a current bit plane, The 1st location list which attached said tag during the clean-up pass of a current bit plane, The 1st location list which did not attach a tag is sorted about a current bit plane. The code changed into the 7th location list and the 8th location list, respectively, The 4th location list which attached said tag during the clean-up pass of a current bit plane, The 4th location list which did not attach a tag is sorted about a current bit plane. The code changed into the 9th location list and the 10th location list, respectively, The code which said location on the said 2nd, 5th, 7th, and 9th lists is merged [be / it / under / clean-up pass / of a current bit plane / comparison], and generates said list [of the location for the next bit planes after a current bit plane] of the 2nd, The code which said location on the said 6th, 8th, and 10th lists is merged [be / it / under / clean-up pass / of a current bit plane / comparison], and generates said list [of the location for the next bit planes after a current bit plane] of the 1st, It has the code which removes the location of the multiplier of the arbitration in Cord Brock which is a multiplier with the significant condition of 1 from said 1st generated list of [for the next bit plane] during the

clean-up pass of a current bit plane.

[0023]

[Embodiment of the Invention] The gestalt of some desirable operations of this invention is explained referring to a drawing below.

[0024] When reference is performed by one or more ones with the same reference number which shows a step and/or the description of accompanying drawings, those steps and/or descriptions have a function or an operation same for the purpose of this explanation, unless a reverse intention appears.

[0025] The principle of the desirable equipment and the approach of a publication has general applicability in this specification to a data compression and defrosting. However, in order to simplify explanation, desirable equipment and a desirable approach are explained, referring to the compression and the defrosting of a digital image according to above-mentioned JPEG2000. However, it does not mean that this limits this invention to the approach and equipment of a publication. For example, this invention can also be applied to compression and defrosting of a digital video.

[0026] Before progressing to the detailed explanation about a desirable approach, an easy general view of a desirable approach is explained below.

First, the whole image is divided into one or more image tile components, and, subsequently two-dimensional dispersion wavelet transform is performed to each of these images tile component. Subsequently, grouping of the transform coefficient of each image tile component is carried out to a subband. Subsequently, this subband is further divided into rectangular Cord Brock, before entropy code modulation of each code Brock is carried out.

[0027] Next, if their eyes are turned to drawing 3 , the flow chart 300 which shows how to carry out entropy code modulation of the symbol of Cord Brock's transform coefficient according to the 1st configuration is shown. Suitably, it is desirable for an approach 300 to call each code Brock of an image. This approach functions by the codec method. That is, this approach can decode or encode a symbol. While either coding or decode is performed, this approach performs the almost same step. decode processing -- SIG -- it becomes coding processing and mirror image relation in that a NIFIKANSU propagation list, a magnitude refinement MENTO list, and a clean-up list are used. However, during the decryption, source code Brock divided into a bit plane does not exist, but the location of the top bit plane is already given. In order to simplify explanation, an entropy-code-modulation method is mainly explained, referring to coding

processing.

[0028] An approach 300 begins from step 302 and one of required parameters is initialized. An approach 300 generates an effective value table among an initialization phase. This effective value table has the two-dimensional array which shows the significant condition of the multiplier of Cord Brock set as the object of coding. First, each entry of an array is set to the zero (0) which show that each multiplier has non-significant condition. The 1st non zero bit plane value of a certain multiplier is encoded, and when the entry to which an array corresponds is reset by (1), the significant condition of the multiplier changes to (1). SIG by which arbitration was called -- under NIFIKANSU propagation coding pass -- or an effective value table can be updated during the clean-up coding pass of arbitration.

[0029] moreover, the inside of an initialization phase and this approach -- this specification -- SIG -- three lists named a NIFIKANSU propagation list (SP list), a magnitude refinement MENTO list (MR list), and a clean-up list (N list) are generated. The location of a bit symbol within the limits of Cord Brock set as the object of coding during current pass is included in these lists. for example, SIG -- a NIFIKANSU propagation list -- Cord Brock's within the limits -- current SIG -- all

the location lists of bit symbols set as the object of coding during NIFIKANSU propagation pass are included. Vocabulary called the location of a bit symbol means the array location of Cord Brock of the transform coefficient in which the bit symbol forms a part within the limits on these specifications. Furthermore, the location of each within the limits of these lists is index-ized according to the order of the Cord Brock scan (refer to drawing 2). the beginning and SIG -- a NIFIKANSU propagation list (SP list) and the maximum refinement MENTO list (MR list) are empty. On the other hand, a clean-up list (N list) lists all Cord Brock's locations within the limits, and is used for coding of the top bit plane with a non zero bit.

[0030] SP list and MR list can be updated during processing of this approach by an addition and deletion of the further location if needed. Furthermore, the present N list is generated during SP coding pass so that the location which is not included in MR list with the present SP may be included.

[0031] Cord Brock's transform coefficient is received after initiation 302, and it is divided into a bit plane (304). It is determined which bit plane has a top non zero bit in it by checking Cord Brock's bit plane at the following step 306.

[0032] Subsequently, this approach progresses to step 308 and entropy code

modulation of the bit plane which has the most significant bit at this step is carried out within single clean-up pass. Entropy code modulation of the bit symbol which specifically belongs to a bit plane with the most significant bit of a non zero bit is carried out respectively. Entropy code modulation of these bit symbols is carried out to drawing 2 in the predetermined order of a scan like illustration. The entropy coding step 308 takes the bit symbol for coding, and its context as an input. This context is suitably determined according to JPEG2000. coincidence -- SIG -- the list of the location set as the object of processing in NIFIKANSU propagation pass and magnitude refinement MENTO pass is generated, consequently the efficient processing of the bit in the next bit plane of this approach is attained. During the first clean-up pass 308, when it becomes clear that a bit symbol is significant, the related location is added to current MR list. This approach is determined also near un-significant [of the inside near 3x3 of the location where the significant thing became clear] during the first clean-up coding pass. Subsequently, these locations are added to current SP list.

[0033] After completion of step 308, this approach progresses to judgment Brock 310, and the check of whether the bit plane set as the object of processing there exists further is performed. If it exists, this approach will progress to step 312

and the bit symbol of the next bit plane will be searched there.

[0034] subsequently, the bit symbol to which this approach relates -- SIG of a current bit plane -- it progresses to step 314 processed with NIFIKANSU propagation pass. At this step 314, entropy code modulation of the bit symbol in the current bit plane corresponding to the location in current SP list is carried out. During SP coding pass 314, when it becomes clear that a bit symbol is significant, it is added to MR list with the new related location. In order to avoid two coding, this new MR list is merged into current MR list after completion of the following clean-up coding pass 318. Moreover, during SP coding pass 314, when it becomes clear that a bit symbol is significant, as for this approach, it determines near un-significant [of the location (P) of the effective-bits symbol]. Subsequently, this approach is added to SP list new near [any] un-significant [of the inside near / that are preceded with the location (P) in order of a scan / these / un-significant]. On the other hand, this approach adds near [any] un-significant [near / that are behind the location (P) in order of a scan / these / un-significant] to current SP list. Subsequently, this approach performs re-index-ization of updated current SP list. This approach also starts generation of N list during SP coding pass 314 using the updated current list SP and current

MR list.

[0035] This approach progresses to the magnitude refinement MENTO coding pass 316 after completion of SP coding pass 314. At this step 316, entropy code modulation of all the bit symbols in the current bit plane corresponding to the location in a magnitude refinement MENTO list is carried out.

[0036] This approach progresses to the clean-up coding pass 318 after completion of MR coding pass 316. At the clean-up coding step 318, entropy code modulation of all the bit symbols in the current bit plane corresponding to the location in a current (generated during the last coding pass) clean-up N list is carried out. During the clean-up coding pass 318, when it becomes clear that a bit symbol is significant, the related location is added to new MR list. This approach is determined also near un-significant [of the location list of / in new MR list]. Subsequently, the location near [these] un-significant is deleted from a clean-up list, and is added to new SP list. Subsequently, new MR list and new SP list are merged into current MR list and SP list, respectively. Subsequently, MR and SP list which were merged newly are formed into a re-index according to the order of a scan.

[0037] After completion of the clean-up coding pass 318, this approach returns

to judgment Brock 310, in order to perform further bit plane processing of arbitration. This approach is ended when judgment Brock returns truth (yes) (i.e., when the further bit plane set as the object of processing does not exist) (300). Suitably, as for this approach, it is desirable to process the bit plane which contains effective bits from the top bit plane to the predetermined lowest bit plane. This approach adds information to the header of the encoded bit stream among a coding phase. The top bit plane and the predetermined lowest bit plane are specified using this information. In a decode phase, this approach uses this header information at the time of a decryption of the encoded bit stream.

[0038] Next, reference of drawing 4 shows the flow chart of the first clean-up coding step 308 used for drawing 3 by the entropy-code-modulation method of illustration. The clean-up coding step 308 starts processing of the top bit plane which has effective bits in it. the time of initiation of the first clean-up coding step 308 -- SIG -- a NIFIKANSU propagation (SP) list and a magnitude refinement MENTO (MR) list are empty, and all multipliers are encoded during this clean-up pass. About each multiplier of this bit plane, when the first clean-up coding step 308 has significant :(1) multiplier which performs the following operations in this bit plane, that multiplier is added to MR list. (2) One near 3x3 of the multiplier of

multipliers is significant, and moreover, when the multiplier itself is not significant, the multiplier is added to SP list.

[0039] It became what has perfect MR list and perfect SP list at the last of the first clean-up coding step 308, and is ready for being used as an object for the following steps. the first clean-up coding step 308 -- a pretreatment step -- these operations are performed by writing further the information about the effective value of the multiplier by which current pretreatment is carried out (412), and information [/ near un-significant / of the arbitration within the limits near 3x3 of a significant multiplier] in memory by encoding each bit symbol of a multiplier in this bit plane 412 inside. While the bit symbol is pretreated so that it may explain to a detail further below, this memory is in the continuation condition of receiving updating. The clean-up coding step 308 also makes a decision about whether a multiplier should be applied to MR list, or it should add to SP list among the after-treatment step 418 by using the information written in memory so that it may explain to a detail further below.

[0040] The first clean-up coding step 308 is a subprocedure, and is called by the main methods 300 of processing the top bit plane which has effective bits in it. The first coding step 308 begins from step 402. Subsequently, as for this

subprocedure, Variables i and j progress to steps 404 and 406 by which both sides are set to 1. Variable j is used as a train counter, Variable i is used as a strip counter, and it enables the first clean-up coding step 308 to process the train and strip of a bit plane of Cord Brock in order of the scan of JPEG2000. Cord Brock has M strip of N train of a bit symbol. The value of M and N is changed according to Cord Brock's size.

[0041] Subsequently, the first clean-up coding step 308 progresses to judgment Brock 408, and the check of whether the strip counter i is below the total of the strip M in 1 or more and Cord Brock is performed here. When judgment Brock 408 returns yes (truth), the first clean-up coding step 308 progresses to judgment Brock 410. Subsequently, judgment Brock 410 confirms whether the train counter j is one or more within Cord Brock, or it is below the total of Train N. When judgment Brock 410 returns yes (truth), the first clean-up coding step 308 progresses to the pretreatment step 412, and this pretreatment step 412 pretreats the j-th train of the i-th strip of that bit plane, and it writes information in memory. It mentions later in a detail further below, referring to drawing 5 about this pretreatment step 412. The first clean-up coding step 308 progresses to step 420 after completion of the pretreatment step 412. When either of judgment

Brock 408 and 410 returns Nor (false), the first clean-up coding step 308 bypasses the pretreatment step 412, and progresses to step 420 directly.

[0042] The first clean-up coding step 308 progresses to judgment Brock 414, the strip counter i is two or more there, and (M+1) the check of whether to be the following is performed to coincidence. However, M is the total of the strip in Cord Brock. When judgment Brock 414 returns yes (truth), the first clean-up coding step 308 progresses to judgment Brock 416, and the check of whether for the train counter j to be three or more, and (N+2) to be the following is performed. However, N is the total of the train in a strip. When judgment Brock 416 returns yes (truth), the first clean-up coding step 308 progresses to the after-treatment step 418 which carries out after treatment of the train of eye watch (j-2) of the strip of eye watch (i-1). The after-treatment step 418 makes a decision about whether a multiplier should be applied to MR list, or it should add to SP list using the information already written in memory. It mentions later in a detail further below, referring to drawing 6 about this after-treatment step 418. The first clean-up coding step 308 progresses to step 420 after completion of the after-treatment step 418. When either returns Nor (false) among judgment Brock 414 and 416, a subprocedure 308 bypasses this after-treatment step 418, and

progresses to step 420 directly. Thus, while the after-treatment step 418 is processing the train of eye watch (j-2) of the strip of eye watch (i-1), the pretreatment step 412 processes the j-th train of the i-th strip.

[0043] The first clean-up coding step 308 carries out the increment of the train counter j only for 1 among step 420. Subsequently, the first clean-up coding step 308 progresses to judgment Brock 422, and the check of whether the train counter j is the following (N+2) there is performed. However, N is the total of the train in a strip. When judgment Brock 422 returns yes (truth), the first clean-up coding pass 308 returns to judgment Brock 408 and 414 who processes the following train j. On the other hand, when judgment Brock 422 returns Nor (false), the first clean-up coding step 308 progresses to step 424, and, as for the strip counter i, the increment only of 1 is carried out there. Subsequently, the first clean-up coding step 308 progresses to judgment Brock 426, and the check of whether the strip counter i is the following (M+1) there is performed. However, M is the total of the strip in Cord Brock. When judgment Brock 426 returns yes (truth), the first clean-up coding step 308 returns to step 406 for the further processing of the following strip.

[0044] As mentioned above, the clean-up coding step 308 reads the information

written in memory, and a decision about whether a multiplier should be applied to MR list or it should add to SP list is made. This memory becomes the form of a current strip register, the last strip register, the following stripline store, the last stripline store, and the last stripline store.

[0045] The last strip register and a current strip register are respectively constituted as a 4xN register arrangement which has four lines and N trains. This array has a 4-N entry in total. This configuration which consists of a front strip register and a current strip register functions as a double buffering configuration. Four lines of a front strip register correspond to four lines which Cord Brock's strip of eye watch (i-1) follows, respectively, and N train of a front strip register is equivalent to N train of the strip [of Cord Brock] of eye watch (i-1). On the other hand, four lines of a current strip register correspond to four lines which Cord Brock's i-th strip follows, respectively, and N train of a current strip register is equivalent to N train of Cord Brock's i-th strip. Each entry of a front strip register and a current strip register supports Cord Brock's multiplier, and has two bits N and S. It is shown [S-bit] whether the multiplier corresponding to the entry is significant. N bit shows whether the multiplier corresponding to the entry exists near 3x3 of a significant multiplier. A current strip register is updated in

pretreatment step 412 middle continuation. For example, S bits of the j-th train of a current strip register are updated during the i-th pretreatment [j-th] 412 of a train of a strip, and N bit which corresponds according to the effective value of the multiplier of the j-th train Cord Brock's i-th strip is updated about a train (j-1), j, and (j+1). The contents of the current strip register are outputted to a front strip register after the i-th completion of pretreatment (inside of step 424) of the train of the last of a strip, and N of a current strip register and S bits are reset by zero. Therefore, the perfect renewal of a current strip register to the i-th strip is included in the strip register in front of under pretreatment of the strip of eye watch (i+1). After treatment 418 has read N and S bits from the front strip register for the strip register i-th current [for strips] in the pretreatment step 412 to coincidence during updating.

[0046] The last stripline store and the following stripline store are respectively constituted as a 1xN Rhine store. This configuration which consists of the last stripline store and the following stripline store functions as a double buffering configuration. The last stripline store corresponds to the 1st line of the i-th strip. The last stripline store is a location which has a significant multiplier in the line of the last of the strip of eye watch (i-1) of the inside near 3x3 of the location, and

stores the row number of the location of the un-significant multiplier in the i-th line [1st / of strip]. The line of the following stripline store corresponds to the 1st line of the strip of eye watch (i+1). The following stripline store is a location which has a significant multiplier in the line of the last of the i-th strip of the inside near 3x3 of the location, and stores the row number of the location of the un-significant multiplier of the 1st line of the strip of eye watch (i+1). As for a row number, in any case, it is stored in the Rhine store one after another from the head of the Rhine store in order of the scan of JPEG2000. Also about which 1 time, the both sides of the following stripline store and the last stripline store can store to N row number. When there is only a row number of under N, the remaining entries of the Rhine store and the last part are left with a blank condition. The following stripline store is updated in pretreatment step 412 middle continuation. For example, any row number of the location of the 1st line of the strip of eye watch (i+1) which is in within the limits near 3x3 of the significant multiplier of the j-th train of the i-th strip during the i-th pretreatment [j-th] 412 of a train of a strip within the limits is added to the following stripline store. The contents of the following stripline store are outputted to the last stripline store after the i-th completion of pretreatment of the train (inside of step

424) of the last of a strip, and the entry of the following stripline store is reset by zero. Therefore, the perfect renewal of the following stripline store to the i-th strip is included at the last stripline store under pretreatment of the strip of eye watch (i+1). The pretreatment step 412 under i-th processing of a strip updates the stripline store next to the strip for [watch (i+1)], and reads the last stripline store of the i-th strip.

[0047] The last stripline store is constituted as a 1x4-line store. The line of the last stripline store corresponds to a train (j-2), (j-1), j, and the line of the last of the strip of eye watch (i-1) of (j+1). The last stripline store is a location which has a significant multiplier in the 1st line of the i-th strip near 3x3 of the location, and stores the row number of the location of the un-significant multiplier of the line of the last of the strip of eye watch (i-1). A row number is stored in order of the scan of JPEG2000. The last stripline store is updated in pretreatment step 412 middle continuation. For example, the row number of the location of the arbitration of the line of the last of the strip of eye watch it is in within the limits near 3x3 of the significant multiplier of the j-th train of the i-th strip (i-1) within the limits is added to the last stripline store during the i-th pretreatment [j-th] 412 of a train of a strip (when it does not exist there beforehand). The after-treatment step 418 has read

the 1st entry of the last stripline store to coincidence. These row numbers are stored one after another in order of the scan of JPEG2000 in the last stripline store of the Rhine store which begins from the head of the Rhine store. Also about which 1 time, the last stripline store can store the row number to 4. When there is only less than four row number, the remaining entries of the last stripline store and the last part are left with a blank condition.

[0048] Next, with reference to drawing 5 , the flow chart of the pretreatment step 412 of the j-th train of the i-th strip which is drawn on drawing 4 explains to a detail further. The pretreatment step 412 begins from step 502, and progresses to step 504. Here, the 1st bit symbol of the j-th train of the i-th strip is searched. This bit symbol for retrieval is the j-th train, and is the first (namely, line 1) bit symbol in order of the scan of JPEG2000 of the i-th strip. Subsequently, the pretreatment step 412 progresses to step 506, and coding and context generation to the bit symbol for retrieval are performed there according to JPEG2000. Step 506 also performs renewal of an effective value table after completion of coding if needed. If the 1st non zero bit plane value of a multiplier is encoded, the significant condition of a multiplier will change to (1) and the entry to which the array of a table corresponds will be reset by (1). When the bit

symbol by which current [belonging to a certain multiplier] was specifically encoded is (1), the effective value of the multiplier stored in an effective value table is set to (1). After completion of step 506, the pretreatment step 412 progresses to judgment Brock 508, and the check of whether the bit encoded there belongs to a significant multiplier is performed. When judgment Brock 508 returns yes (truth), the pretreatment step 412 progresses to step 510, it is the j-th train corresponding to the bit symbol for retrieval there, and the bit S of the j-th train of a current strip register is set to (1). When judgment Brock 508 returns Nor (false), the pretreatment step 412 progresses to judgment Brock 534 directly (namely, when a multiplier is not significant).

[0049] After step 510, the pretreatment step 412 progresses to loop formations 514-526, and it is processed near [each] ** significant [of the inside near 3x3 of a current significant multiplier] in order there. The pretreatment step 412 acquires near un-significant [of a current significant multiplier] among step 514 (when it exists). Step 514 is determined near un-significant [of a current significant multiplier] by investigating an effective value table. If it does not exist near un-significant, the pretreatment step 412 progresses to judgment Brock 534 directly (not shown). On the other hand, when it exists near un-significant, the

pretreatment step 412 progresses to judgment Brock 516, and the check of whether then, current near is in the last strip (i-1) is performed. Then, when judgment Brock 516 returns yes (truth), the pretreatment step 412 progresses to step 518, and the row number of current near is added at the last stripline store. The pretreatment step 412 progresses to judgment Brock 526 after step 518. On the other hand, when judgment Brock 516 returns Nor (false), the pretreatment step 412 progresses to judgment Brock 520, and the check of whether current near is in the following strip (i+1) there is performed. Then, when judgment Brock 520 returns yes (truth), the pretreatment step 412 progresses to step 522, and a row number is added at the following stripline store. The pretreatment step 412 progresses to judgment Brock 526 after step 522. On the other hand, when judgment Brock 520 returns Nor (false), the pretreatment step 412 progresses to step 524. The pretreatment step 412 sets Bit N to (1) within the entry of the current strip register corresponding to current near among step 524. The pretreatment step 412 progresses to judgment Brock 526 after step 524. Judgment Brock 526 confirms whether near which is not processed yet by loop formations 514-526 exists in the inside near 3x3 of a current significant multiplier further. When judgment Brock 526 returns yes (truth), in order to process near

the degree of a current significant multiplier, the pretreatment step 412 returns to step 514. When judgment Brock 526 returns Nor (false), a pretreatment step progresses to step 528.

[0050] Among step 528, the pretreatment step 412 searches the head of the last stripline store which is the 1st entry of the last stripline store which follows in order of the scan of JPEG2000, and assigns it to Variable headSP. Among the i-th pretreatment step 412 of the beginning of a strip (namely, j= 1), the last stripline store corresponds to the 1st line of the i-th strip, and, subsequently stores the row number of a location with the significant multiplier in the strip of eye watch (i-1) near [those] 3x3. Therefore, the 1st row number stored in the variable headSP in the i-th last stripline store of the 1st line of a strip of a certain location in order of the scan of JPEG2000 is contained among the i-th pretreatment step 412 (namely, j= 1) of the beginning of a strip. This location has a significant multiplier near 3x3 of that location in the strip of eye watch (i-1). After step 528, the pretreatment step 412 progresses to judgment Brock 530, and the check of being j=headSP is performed there. When judgment Brock 530 returns yes (truth), the pretreatment step 412 progresses to step 532, and N bit is set to (1) there as an object for the entries of the current strip register

corresponding to a location (1 j). Thus, when setting those corresponding N bits near un-significant [of the i-th strip of the significant multiplier in the strip of eye watch (i-1)], it is specified in a current strip register. Moreover, the current head of the last stripline store is removed by the pretreatment step 412 among step 532. Furthermore, all the remaining row numbers are shifted to the head of the last stripline store at once. The pretreatment step 412 progresses to judgment Brock 534 after completion of step 532. On the other hand, when judgment Brock 530 returns Nor (false), the pretreatment step 412 progresses to judgment Brock 534 directly. The last stripline store stores the row number to N in this contractor first so that clearly. The pretreatment step 412 confirms [each train j= 1, --,] whether if called about N, a row number [step / 412 / pretreatment / between judgment Brock 530 and in the last stripline store] exists. If a corresponding row number exists, it will be specified near un-significant as what exists in the 1st line of the train, it will be inputted into a current strip register, and will be removed from the last stripline store.

[0051] Judgment Brock 534 judges whether the bit symbol which is not processed yet by the pretreatment step 412 exists further in the j-th train. When judgment Brock 534 returns yes (truth), the pretreatment step 412 progresses to

step 536, and the following bit symbol within a train is searched there. Subsequently, the pretreatment step 412 returns to step 506 which processes this searched bit symbol. On the other hand, if the bit symbol which should be processed more than it does not exist in Train j, judgment Brock 534 returns Nor (false), and it ends (538), and the pretreatment step 412 returns to the first clean-up coding step 308, in order to perform further processing.

[0052] Next, if an eye is changed to drawing 6 , the flow chart of the after-treatment step 418 of the train of eye watch (j-2) of the strip of eye watch (i-1) it is drawn on drawing 4 is further shown in the detail. As mentioned above, while the pretreatment step 412 is processing the j-th train of the i-th strip, as for the after-treatment step 418, the train of eye watch (j-2) of the strip of eye watch (i-1) is processed.

[0053] The after-treatment step 418 begins from step 602, it progresses to step 604, and a variable position1 is set to (1, j-2) there. This variable position1 stores the coordinate (a line, train) of the current position in the strip register before being processed by the after-treatment step. After completion of step 604, it progresses to step 610, the entry then, stored at the head of the last stripline store is searched, and the after-treatment step 418 is stored in Variable headSP.

This variable headSP stores the row number of a location in the strip of eye watch which has a significant multiplier in the i-th strip near 3x3 of a certain location (i-1). When the last stripline store is empty, step 606 sets Variable headSP to null.

[0054] After completion of step 606, the after-treatment step 418 progresses to judgment Brock 608, and the check of being variable headSP= (j-2) is performed there. When judgment Brock 608 returns yes (truth), the after-treatment step 418 progresses to step 610, and a variable positon2 is set to (4, headSP) there. This variable positon2 shows (the line and train) of the bit symbol in the strip of eye watch (i-1), and this symbol has a significant multiplier near [that] 3x3 in the i-th strip. When judgment Brock 608 returns Nor (false), this variable positon2 is set to null (612). The after-treatment step 418 progresses to step 614 after completion of step 612.

[0055] Among step 614, the after-treatment step 418 compares variables position1 and positon2, and it judges whether position1 exists before positon2 in order of the scan of JPEG2000. If it exists, the after-treatment step 418 stores a variable position1 in variable winning_position. If it does not exist, the after-treatment step 418 stores a variable positon2 in variable winning_position.

After completion of step 614, the after-treatment step 418 progresses to judgment Brock 616, and the check of whether the coordinate stored in variable winning_position there is a variable positon2 (namely, the last stripline store) is performed.

[0056] When judgment Brock 616 returns yes (truth), an after-treatment step progresses to step 618, and the head of the last stripline store corresponding to headSP is removed there. Moreover, as for the remaining entries of the last stripline store, only 1 is respectively shifted to a head. An after-treatment step progresses to step 620 after completion of step 618, and N bit in the location in the strip register before corresponding to the location stored in variable winning_position there is set to (1). The after-treatment step 418 progresses to judgment Brock 622 after completion of step 620. When judgment Brock 612 returns Nor (false), an after-treatment step progresses to judgment Brock 622 directly.

[0057] Judgment Brock 622 confirms whether S bits of the location in the strip register before corresponding to the location stored in variable winning_position are (1). When judgment Brock 622 returns yes (truth), the after-treatment step 418 progresses to step 624, and the location stored in variable winning_position

there is added to MR list. The after-treatment step 418 progresses to judgment Brock 630 after completion of step 624. When judgment Brock 622 returns Nor (false), the after-treatment step 418 progresses to judgment Brock 626. Judgment Brock 626 confirms whether N bit of the location in the strip register before corresponding to the location stored in variable winning_position is (1). When judgment Brock 626 returns yes (truth), the after-treatment step 418 progresses to step 628, and the location stored in variable winning_position there is added to SP list. The after-treatment step 418 progresses to judgment Brock 630 after completion of step 628. When judgment Brock 626 returns Nor (false), the after-treatment step 418 progresses to judgment Brock 630 directly.

[0058] Judgment Brock 630 confirms whether the location set as the object of processing exists further in the train of eye watch (j-2). That is, judgment Brock 630 confirms whether the line of the current position stored in variable winning_position is equal to 4. When judgment Brock 630 returns Nor (false), the after-treatment step 418 progresses to step 632, and, as for the line number in a variable position1, the increment only of (1) is carried out there. After step 632, the after-treatment step 418 progresses to step 606, and after treatment 418 is resumed to the new value stored in the variable position1 there. On the other

hand, when judgment Brock 630 returns yes (truth), it ends (634), and the after-treatment step 418 returns to the clean-up coding step 308, in order to perform further processing.

[0059] Since it stores in a current strip register, the pretreatment step 412 to the i-th strip generates current N and S bits, so that the above may show. It exists as i-th object for strips at the time of termination of a pretreatment step, and, subsequently such N and S bits are outputted to a front strip register. The after-treatment step 418 processes N of the strip of eye watch current storing is carried out (i-1), and S bits to a front strip register at the pretreatment step 412 and coincidence of the i-th strip. This N and S bits are generated as last object for the strips of eye pretreatment step 412 Naka (i-1) watch. The pretreatment step 412 of the i-th strip pinpoints near [in the i-th strip] a significant multiplier and near [its] un-significant, and stores them as N of the location where a current strip register corresponds, and S bits. The pretreatment step 412 specifies near [any] un-significant [of the 1st line of the i-th strip of the significant multiplier in the strip of eye watch (i-1)], and stores these in the location where a current register corresponds as an N bit. The pretreatment step 412 of the i-th strip processes the latter by searching the row number such near

un-significant [of the arbitration by which current storing is carried out] at the last stripline store. It is specified near [these] un-significant at the last pretreatment step of the strip of eye watch (i-1), they are stored in the following stripline store, and are outputted to the last of the pretreatment step of the strip of eye watch (i-1) at the last stripline store. By the pretreatment step 412 of the i-th strip, it is near un-significant [resulting from the significant multiplier in the i-th line / 1st / of strip], and is specified near [any] un-significant [in line of last of strip of eye watch (i-1)]. The pretreatment step 412 adds the row number near [these] un-significant to the last stripline store. Subsequently, the after-treatment step 418 to the strip of eye watch (i-1) searches these row numbers out of the last stripline store, and updates N bit to which the arbitration in a front strip register corresponds. (i-1) The after-treatment step 418 to the strip of eye watch pinpoints the location which can be added on SP list or MR list using N stored in the strip register before being updated, and S bits. The location in the strip register before [which specifically has S bits set to 1] being updated is stored in MR list, and the location in the strip register before [with S= 1 and N= 0] being updated is stored in SP list.

[0060] If it returns to drawing 3 , after completion of the first clean-up coding step

308, this approach will progress to judgment Brock 310, and the check of whether the bit plane set as the object of processing there exists further will be performed. If it exists, this approach will progress to step 312 and the bit symbol of the next bit plane will be searched there. subsequently, this approach -- SIG of a current bit plane -- it is NIFIKANSU propagation pass and progresses to step 314 which processes a related bit symbol.

[0061] next -- if an eye is changed to drawing 7 A and drawing 7 B -- SIG of drawing 3 -- the flow chart of the NIFIKANSU (propagation SP) coding step 314 is further shown in the detail. This coding step 314 follows the clean-up coding step 318 for each bit planes after an after that and first stage bit plane following the first clean-up coding step 308 at first. SP coding step 314 is called the entropy-code-modulation method 300, and takes the form of a subprocedure. This coding step 314 begins from the place passed to SP coding step 314, in order that current SP list (it is henceforth called original SP list on these specifications) may process (702). This original SP list is generated by the first last clean-up step 308 or last clean-up coding step 318. SP coding step 314 uses original SP list passed to SP coding step 314, the near SP list, the strip SP list of degrees, the last strip SP list, and five lists of new SP lists. The four latter

lists are generated by SP coding pass 314. SP list, the last strip SP list, and the strip SP list of degrees are local at the point that use of those lists is limited to SP coding step 314 soon. new SP list generated on the other hand and updated original SP list -- a step -- in order to perform further processing 318 inside, an entropy-code-modulation method is passed.

[0062] Since the multiplier of the arbitration corresponding to the location stored in this list which is a multiplier which becomes significant by SP coding step 214 present Nakaato is specified by tagging, SP list of SP coding step 314 middle original can be updated. Each entry of SP list original for this purpose is a location which has a significant multiplier near that location 3x3, and has the coordinate which shows the location of the bit symbol in the current bit plane of an un-significant multiplier (a line, train), and the flag which shows the effective value of a bit symbol after current SP coding step 214.

[0063] New SP list has the coordinate (a line, train) of the location of the bit symbol of the un-significant multiplier which has the new significant multiplier of arbitration near 3x3 of the location. These new significant multipliers are encoded significant multipliers which have a bit symbol in a current bit plane, and the significant condition of these significant multipliers changes to one

among current SP coding step 314 (it changes to owner mind). since the multiplier of the arbitration corresponding to the location stored in this list which is a multiplier which becomes significant by SP coding step 314 present Nakaato is specified by tagging -- SP coding step -- new SP list can be updated 314 inside. Each entry of SP list new for this purpose is a location which has a new significant multiplier near 3x3 of that location, and has the coordinate (a line, train) of the location of the bit symbol of an un-significant multiplier, and the flag which shows the effective value of a bit symbol after current SP coding step 314.

[0064] SP list will have the coordinate (a line, train) of a location within the limits of Cord Brock of some bit symbols soon. SP list is the subset of new SP list soon. That is, all the locations in SP list will be included in this new list soon. The location (however, they are necessarily no locations) of an un-significant multiplier will be included in within the limits near 3x3 of the current significant multiplier just decrypted among SP coding step 314 by SP list soon. Only the location within the limits near [which has SP list in the strip of the same Cord Brock as a current significant multiplier, and is after a significant multiplier current in the order of a scan of JPEG2000] 3x3 will be stored soon.

[0065] the strip SP list of degrees -- SP coding step -- it has the train coordinate

of the location of an un-significant multiplier within the limits of the 1st line of the next strip within the limits near [where the thing significant 314 inside became clear] 3x3 of the multiplier in a current strip. The strip SP list of degrees is the subset of new SP list at the point of being contained in a list with all the locations new soon in SP list.

[0066] the last strip SP list -- SP coding step -- it has the train coordinate of the location of an un-significant multiplier within the limits of the 1st line of the current strip within the limits near [where the thing significant 314 inside became clear] 3x3 of the multiplier in the last strip.

[0067] After initiation, SP coding step 314 progresses to the initialization step 704, and the entry to SP list, the strip SP list of degrees, the last strip SP list, and new SP list will be reset by zero (0) there soon. After the initialization step 704, a subprocedure progresses to judgment Brock 706 and the check of whether then, a certain location exists in original SP list is performed. When judgment Brock 706 returns Nor (false), SP coding step 314 is ended (714) and an entropy-code-modulation method progresses to MR coding step 316. When all Cord Brock's locations within the limits are significant and are set as the object of coding among MR coding step 316, the above-mentioned situation may arise.

When judgment Brock 706 returns yes (truth), SP coding step 314 progresses to a loop formation (steps 710-728) and the clean-up list generation step 708 at coincidence. While SP coding step 314 encodes SP location and is generating above-mentioned SP list, the clean-up list generation step 708 generates a clean-up list. The clean-up list generation step 708 is explained to a detail from the following.

[0068] A loop formation (710-728) processes each location in SP list original about each pass of a loop formation, the near SP list, and the last strip SP list. The location of a degree which SP coding step 314 is the next location in original SP list, the near SP list, and the last strip SP list, and was not searched during the last pass during the pass of the arbitration of a loop formation is searched (720). The location in these lists is searched one after another during the pass of a loop formation according to the order of a scan of JPEG2000. It mentions later in a detail further below, referring to drawing 8 about the search method of these locations. On the other hand, when the location which was not before searched within the limits of these lists does not exist any more, SP coding step 314 exits from a loop formation (steps 710-728), and progresses to step 712.

[0069] After completion of the initialization step 704, SP coding step 314

progresses to judgment Brock 706, and the check of whether then, a location exists further in original SP list, the near SP list, and the last strip SP list is performed. If the remaining location exists within the limits of these lists, judgment Brock will return yes (truth) and SP coding step 314 will progress to judgment Brock 710.

[0070] Judgment Brock 716 confirms whether the location which was not searched before exists further in Cord Brock's current strip, original SP list, the near SP list, and the last strip SP list. SP coding step 314 processes Cord Brock's strip from the 1st strip, is the order of a scan indicated by JPEG2000, and continues processing all the time to the last strip. A current strip is Cord Brock's 1st strip during the pass of judgment Brock's 716 beginning. When judgment Brock 716 returns Nor (false), SP coding step 314 progresses to step 718, the aisle location in the strip SP list of degrees is outputted to the last strip SP list there, and the entry of the strip SP list of degrees is set to zero. it mentioned above -- as -- the strip SP list of degrees -- SP coding step -- it has the train coordinate of the location of the un-significant multiplier of the 1st line of the next strip within the limits near 3x3 of the multiplier in a current strip which is a multiplier the thing significant 314 inside was proved that it is within the limits.

the last strip SP list -- SP coding step -- it has the train coordinate of the location of the un-significant multiplier of the 1st line of the current strip within the limits near 3x3 of the multiplier in the last strip which is a multiplier the thing significant 314 inside was proved that it is within the limits. SP coding step 314 adds a row number to the strip SP list of degrees (step 738), and searches a row number out of the last strip SP list during processing of the strip SP coding Cord Brock's arbitration. If a location does not exist within the limits of a current strip any more, the row number of the strip SP list of degrees is outputted to the last strip SP list (718), the entry of the strip SP list of degrees is set to zero, and the following strip turns into a current strip. SP coding step 314 progresses to step 720 after completion of step 718.

[0071] On the other hand, when judgment Brock 716 returns yes (truth), SP coding step 314 progresses to step 720 directly. Among step 720, SP coding step 314 is the next location in original SP list, the near SP list, and the last strip SP list, and searches the location which was not searched during the pass of the last time of a loop formation. It explains to a detail more, referring to drawing 8 about this below. After completion of step 720, SP coding step 314 progresses to step 722, and context generation and coding of a bit symbol are performed in

the (retrieval) location there. This coding and context generation are suitably performed according to JPEG2000. After completion of coding, step 722 also performs renewal of an effective value table, when required. The 1st non zero bit plane value of a multiplier is encoded, and when the entry to which the array of a table corresponds is reset by (1), the significant condition of a multiplier changes to (1). When the bit symbol by which current [belonging to a certain multiplier] was specifically encoded is (1), the effective value of the multiplier stored in an effective value table is set to (1).

[0072] SP coding step 314 progresses to judgment Brock 724 after completion of the coding step 722. In judgment Brock 724, the check of whether the encoded bit symbol in the (retrieval) location has the significant condition of (1) is performed. When judgment Brock 724 returns Nor (false), the location of a degree where SP coding step 314 is [in] the next location in SP list original return and there, the near SP list, and the last strip SP list to step 710, and was not searched during the pass of the last time of a loop formation is searched. On the other hand, when judgment Brock 724 returns yes (truth), SP coding step 314 progresses to step 726, and a tag is attached as a thing effective in the flag of a location (searched) with which it corresponds in an original list or new SP list

there. Thus, among the following clean-up coding step 318, in order to perform the addition on MR list which follows, this location is pinpointed.

[0073] SP coding step 314 passes new SP the coordinate (a line, train) near [this] un-significant after completion of step 726 to the processing for adding near un-significant (732). It mentions later in a detail further, referring to drawing 13 A and drawing 13 B about this. This processing begins from initiation of SP coding step 314, and is ended by completion of SP coding step 314.

[0074] after completion of step 726, and SP coding step 314 -- subloop 728- it progresses to 738 and 740 and the un-significant multiplier of the inside near the 3x3 of a current significant multiplier which has the bit symbol by which current was encoded there is processed in order about each pass of a subloop. subloop 728- 738 and 740 have judgment Brock 728, it is this judgment Brock and the check of whether to exist near un-significant [of a current significant multiplier which was not before processed by the subloop] further is performed. Judgment Brock 728 performs this judgment with reference to an effective value table. When judgment Brock 728 returns Nor (false), SP coding step 314 returns to judgment Brock 710, in order to perform further processing. On the other hand, when judgment Brock 728 returns yes (truth), SP coding step 314 progresses to

step 730, and the location near [as a coordinate (a line, train)] un-significant is searched from an effective value table there. After completion of step 730, SP coding step 314 progresses to judgment Brock 734, and the check of whether the location near [there current] un-significant is behind the location of the multiplier which has the bit symbol by which current was encoded in order of the scan of JPEG2000 is performed. When judgment Brock 734 returns Nor (false), in order to perform further processing, SP coding step 314 returns to judgment Brock 728. On the other hand, when judgment Brock 734 returns yes (truth), SP coding step 314 progresses to judgment Brock 736, and the check of whether to belong near [current] un-significant to the following strip is performed after the current strip of the phase hand to whom the bit symbol by which current was encoded there belongs. When judgment Brock 736 returns yes (truth), SP coding step 314 progresses to step 738, and the row number of the location near [current] un-significant is added at the last of the strip SP list of degrees there. After completion of step 738, SP coding step 314 returns to judgment Brock 728, in order to perform further processing. On the other hand, when judgment Brock 738 returns Nor (false), SP coding step 314 progresses to step 740, and the coordinate (a line, train) of the location near [current] un-significant will be

added at the last of SP list there soon. After completion of step 740, SP coding step 314 returns to judgment Brock 728, in order to perform further processing.

[0075] Next, if an eye is changed to drawing 8 , the flow chart of step 720 of SP coding step 314 is further shown in the detail. According to the order of a scan of JPEG2000, step 720 is the 1st location stored in original SP list, the near SP list, or the last strip SP list, and searches the 1st location which was not searched and processed before. As mentioned above, the location stored in original SP list, the near SP list, and the last strip SP list is sorted according to JPEG2000 so that clearly. Three head pointers have pointed out the 1st entry to original SP list, the 1st entry to the near SP list, and the 1st entry to the last strip SP list during the 1st [to step 720] call, respectively. About the each call which follows, one in these head pointers moves to the next item in a list. When step 718 outputs the contents of the strip SP list of degrees to the last strip SP list, the head pointer for the last strip SP list is reset by the 1st entry of the last strip SP list. The location of a head pointer is held between each call to step 720 in the current entropy step 314. A head pointer will be reset shortly after the current entropy step 314 is completed.

[0076] Step 720 begins from step 802 and progresses to judgment Brock 804,

806, and 808 at coincidence. By judgment Brock 804, step 720 confirms whether the head pointer has pointed out the significant (that is, it is not null) location in original SP list. When judgment Brock 804 returns yes (truth), step 720 progresses to step 812, and step 720 searches the coordinate (a line, train) of that location in original SP list which the head pointer has pointed out, and it stores this coordinate in Variable origSP there. On the other hand, when judgment Brock 804 returns Nor (false), step 720 sets Variable origSP to null. By judgment Brock 806, step 720 confirms whether the head pointer has pointed out the location significant (that is, it is not null) within SP list soon. When judgment Brock 806 returns yes (truth), step 720 progresses to step 816, and step 720 searches the coordinate (a line, train) of that location in the near SP list which the head pointer has pointed out, and it stores this coordinate in Variable neighSP there. On the other hand, when judgment Brock 806 returns Nor (false), step 720 sets Variable neighSP to null. It is confirmed whether, by judgment Brock 808, step 720 has pointed out the location where a head pointer is significant in the last strip SP list (that is, it is not null). When judgment Brock 808 returns yes (truth), step 720 progresses to step 820 and step 720 searches the row number of the location in the last strip SP list which the head pointer has

pointed out there. This row number is stored as a coordinate in Variable nextstripSP (1 row number). As mentioned above, the last SP list is near un-significant [resulting from the significant multiplier encoded among SP coding step 314 of the last strip], and stores the row number near un-significant [of the arbitration of the 1st line of a strip]. On the other hand, when judgment Brock 808 returns Nor (false), step 720 sets Variable nextstripSP to null.

[0077] Step 720 progresses to step 822 after completion of steps 810-820. The location of the location stored in Variables origSP, neighSP, and nextstripSP is compared among step 822, it is the location which consists of these three locations, and the location concerned set to the 1st in the order of a scan of JPEG2000 is discovered. After completion of step 822, step 720 progresses to step 824 and only one entry is moved in the direction distant from this 1st entry within the list with which the head pointer to the entry which includes this '1st' location there corresponds. Subsequently, this '1st' location obtained by the comparison step 822 is passed to the coding step 722 of SP coding step 314, and the clean-up list generation step 708, in order to perform further processing. Subsequently, step 720 is ended.

[0078] While SP coding step 314 is carrying out steps 710-740 during activation,

SP coding step 314 performs the clean-up list generation step 708. This is explained to a detail from the following.

[0079] Next, if an eye is changed to drawing 13 A and drawing 13 B, the flow chart which shows the processing which adds near un-significant to new SP list is shown. SP coding step 314 -- SP coding pass -- the coordinate (a line, train) near un-significant of the multiplier the thing significant 314 inside was proved that it is is passed into step 732 (drawing 7 A). These locations are passed to the processing 1300 for adding near un-significant to new SP list. This processing 1300 begins from initiation of SP coding step 314, and is ended by completion of SP coding step 314. The purpose of this processing 1300 is removing the overlap part of the arbitration near un-significant [of a significant multiplier]. For example, since the multiplier which it is near [which consists of two significant multipliers] un-significant exists, an addition on new SP list may be needed only once. It is added near [these] un-significant to new SP list, and sorting is performed within SP list in order of the scan of JPEG2000. Processing 1300 uses the strip register in front of 4xN, the current strip register of 4xN, the last stripline store of 1xN, the stripline store next to 1xN, and the last stripline store of 1x4 mentioned above. However, N is Brock's width of face. Processing 1300

functions on a pretreatment step which was used with the first clean-up coding pass 308 as a pretreatment step by the somewhat similar approach. Processing 1300 sends the contents of the last strip register and the last stripline store to the after-treatment process 1400. About this, it mentions later in a detail further with reference to drawing 14 . A treatment process 1400 functions as an after-treatment step after this by the approach used with the first clean-up coding pass 308, and the somewhat similar approach. For example, while processing 1300 is pretreating the i-th strip, processing 1400 will carry out after treatment of the strip of eye watch (i-1).

[0080] A pretreatment process 1300 sets to (1) N bit of the entry of the current strip register of 4xN for un-significant multipliers with which it corresponds in the i-th same strip which is the un-significant multiplier which exists in within the limits near 3x3 of the significant multiplier in the i-th strip. A pretreatment process 1300 stores the row number of the un-significant multiplier which is in the line of the last of the strip of eye watch (i-1), and is near 3x3 of the significant multiplier of the 1st line of the i-th strip again in the last stripline store. once the i-th strip is pretreated -- this -- the i-th strip is sent to processing 1400, in order to perform after treatment. Processing 1400 is an entry in the last stripline store, and

fetches the entry by which the current addition is carried out to the last stripline store during pretreatment of the strip of eye watch (i+1) again. That is, the process 1400 which carries out after treatment fetches in order the row number of the un-significant multiplier which is in the i-th line of the last of a strip, and (i+1) is near 3x3 in the significant multiplier of the 1st line of the strip of eye watch.

[0081] Processing 1300 begins from step 1302 and the entry of an above-mentioned register and the Rhine store are set to zero there. Subsequently, processing 1300 progresses to step 1304 and the fetch of the coordinate (a line, train) of the significant multiplier then, passed into step 732 is carried out. This coordinate (a line, train) by which the fetch was carried out is stored in Variable CurPos. Subsequently, processing 1300 progresses to judgment Brock 1306, and the check of whether an un-significant multiplier exists in the inside near the 3x3 of a location stored in CurPos there is performed.

[0082] When judgment Brock 1306 returns yes (truth), processing 1300 progresses to loop formations 1312-1324, the fetch of near un-significant [of the inside near 3x3 of a location CurPos] is carried out to order, and it is processed there. Processing 1300 progresses to step 1312, and the fetch of the coordinate

(a line, train) near [next] un-significant is carried out there, and, specifically, it is stored in Variable NextN. Subsequently, processing 1300 progresses to judgment Brock 1314, and the check of whether then, a location NextN exists in the strip in front of a current strip is performed. When judgment Brock 1314 returns yes (truth), the row number of a location NextN is added to the last stripline store (1316). When yes (truth) is not returned, processing 1300 progresses to judgment Brock 1320, and the check of whether then, a location NextN exists in the next strip of a current strip is performed. When judgment Brock 1320 returns yes (truth), the row number of a location NextN is added to the following stripline store (1318). On the other hand, when judgment Brock 1320 returns Nor (false), processing 1300 sets Bit N to (1) to the entry in the current register corresponding to a location NextN. Processing 1300 progresses to judgment Brock 1324 after steps 1316 and 1318 or completion of 1322. By judgment Brock 1324, the check of whether to exist near un-significant further is performed. When judgment Brock 1324 returns yes (truth), processing 1300 performs further processing return and near [next] un-significant to step 1312. When yes (truth) is not returned, a step continues to 1328.

[0083] Variable PrevPos is set to Variable CurPos at step 1328. Subsequently,

processing 1300 continues to step 1330 and processing 1300 fetches the coordinate (a line, train) of the location of a degree passed into step 732 (drawing 7 A) there. This coordinate (a line, train) by which the fetch was carried out is stored in Variable CurPos. When this location of a degree still is not available, processing 1300 stands by until it becomes available. If an available location does not exist any more, processing 1300 progresses to step 1334 directly (not shown).

[0084] After completion of step 1330, processing progresses to judgment Brock 1332 and the check of whether the location then, stored in Variable CurPos exists in the following strip about the location stored in Variable PrevPos is performed. When judgment Brock 1332 returns Nor (false), further processing of the new location where processing 1300 was stored in judgment Brock 1306 in return and Variable CurPos is performed. When judgment Brock 1332 returns yes (truth), processing 1300 continues to step 1334.

[0085] Among step 1334, processing 1300 moves the contents of the current strip register to a front strip register, and sets the entry in a current strip register to zero. Step 1334 moves the contents of the following stripline store to the last stripline store, and, subsequently to zero, sets the entry in the following stripline

store. Processing follows coincidence to steps 1336 and 1338 after completion of step 1334. Processing 1300 carries out after treatment of the front strip register among step 1336. It explains to a detail further, referring to drawing 14 below about this. Processing 1300 sets Bit N to (1) among step 1338 as an object for the entries in the current strip register corresponding to the location by which current storing is carried out into the last stripline store. After completion with steps 1336 and 1338, processing continues to judgment Brock 1306, in order to process near un-significant further.

[0086] When judgment Brock 1306 returns Nor (false) (i.e., when it does not exist near un-significant), processing 1300 continues to judgment Brock 1308. Judgment Brock 1308 confirms whether the location passed by step 732 (drawing 7 A) exists further. When judgment Brock 1308 returns yes (truth), processing 1300 progresses to step 1304 and the fetch of the location is carried out there. When a location available now does not exist, judgment Brock 1308 stands by until a location is generated, or SP coding pass 314 stands by until SP coding pass 314 is completed at the end time (1310) which processing ends.

[0087] Next, if an eye is changed to drawing 14 , the flow chart of the after treatment 1400 used with the pretreatment 1300 of drawing 13 A and 13B is

shown. The after-treatment step 1400 starts in SP coding pass 314 and coincidence. After treatment 1400 starts 1402 and the strip register before after treatment 1400 is passed into step 1336 is stood by (1404). After a front strip register is passed to after treatment 1400, processing 1400 progresses to loop formations 1406-1416. Loop formations 1406-1416 are each entry in a front strip register (namely, location), they process in order each entry with N bit set to (1), compare with its entry the entry (row number) by which current storing is carried out into the last Rhine store, and add the location before the order of a scan of JPEG2000 to the new list SP. A loop formation 1406 processes the entry in a front strip register in order of the scan of JPEG2000. Pretreatment 1300 has updated the last Rhine store also for under after treatment 1400 continuously by adding an entry so that clearly from the above.

[0088] Loop formations 1406-1416 have judgment Brock 1406 and step 1408. By judgment Brock 1406, processing 1400 confirms whether a certain location exists a certain entry (namely, location) or in the last stripline store in the strip register before having N bit set to (1). When judgment Brock 1406 returns yes (truth), processing 1400 progresses to steps 1408 and 1410 at coincidence. At step 1408, processing 1400 fetches the following row number (the 1st) in the

Rhine store of current last time, and stores this row number in Variable LinePos. Variable LinePos is set to null if a current entry does not exist in the last Rhine store. Step 1410 fetches the location with N bit which is the location of a degree by which a fetch was not carried out and was set before (1) of a degree from a front strip register in order of the scan of JPEG2000, and stores the location in Variable StripPos at coincidence. After steps 1408 and 1410, processing 1400 progresses to step 1412 and processing compares the location stored in Variables LinePos and StripPos there. Step 1412 is this location stored in Variables LinePos and StripPos, determines this location located in order of the scan of JPEG2000 in an early location, and stores the location in Variable WinPos. When then, processing progresses to step 1414 after completion of step 1412 and the location exists in the last Rhine store, the location stored in Variable WinPos is removed from the last Rhine store. Processing progresses to step 1416 after completion of step 1416, and the location stored in Variable WinPos there is added to new SP list. In order that processing 1400 may perform further processing, it returns to judgment Brock 1406. Processing 1400 is ended when judgment Brock 1406 returns Nor (false) (1418).

[0089] Next, if an eye is changed to drawing 9 , the flow chart of the clean-up list

generation step 708 of SP coding step 314 is shown. The clean-up list generation step 708 begins from step 902, progresses to step 904, and acquires the location defined at step 720 during the 1st [of a loop formation] pass (710-740) there. This location is the 1st location which follows in order of the scan of JPEG2000, and this location is stored in either original SP list, the near SP list or the last strip SP list. This location is stored in Variable headSP. Moreover, step 904 acquires the 1st location stored in MR list according to the order of a scan of JPEG2000, and stores this location in Variable headMR. The clean-up generation step 708 progresses to a loop formation 906 after completion of step 904.

[0090] A loop formation 906 carries out the increment only of 1 through each location P of Cord Brock (x y) in the order of a scan of JPEG2000. A loop formation 906 progresses to judgment Brock 908 after each increment over Cord Brock's new location P (x y). Judgment Brock 908 confirms whether the current position P of a loop formation (x y) is equal to the location stored in Variable headSP. When judgment Brock returns Nor (false), the clean-up generation step 708 progresses to judgment Brock 912, and a check equal to the location where the current position P of a loop formation (x y) was stored in Variable headMR

there is performed. When judgment Brock returns Nor (false), the clean-up generation step 708 progresses to step 916, and the current position P (x y) is added to a clean-up list there. After completion of step 916, it is carried out [to a loop formation 906] return and there, and the increment of the location P (x y) is carried out for the clean-up generation step 708 in the order of a scan of JPEG2000 in the location of a degree.

[0091] On the other hand, when judgment Brock 908 returns yes (truth), the clean-up generation step 708 progresses to step 910. Among step 910, when the location of a degree defined at step 720 is available, the clean-up generation step 708 acquires the location, and stores the location in Variable headSP. Thus, the clean-up generation step 708 replaces with the location of a degree the current position stored in Variable headSP whenever it searched the location defined during the pass of step 720 one after another and called step 910. There is sometimes a case where the current use of the location of a degree cannot be carried out. In such a situation, the clean-up generation step 708 stands by until the location becomes available. Furthermore, when judgment Brock 710 (drawing 7 A) returns Nor (false), the clean-up generation step 708 sets Variable headSP into this step 910 at null instead of standing by. In the case of the latter,

the location set as the object of coding among SP coding step does not exist any more. However, the location set as the object of coding which needs to be added to a clean-up list among SP coding step may exist behind the last location. By setting Variable headSP to null, these locations can be added to a clean-up list during the consecutiveness pass of a loop formation 906. A clean-up generation step is carried out [to a loop formation 906] after completion of step 910 return and there, and, as for a location P (x y), the increment only of 1 is carried out in the order of a scan of JPEG2000.

[0092] Furthermore, when judgment Brock 912 returns yes (truth), the clean-up generation step 708 progresses to step 914, and the next location in MR list is stored in Variable headMR there. Thus, a clean-up generation step searches the location in MR list one after another, and replaces with the location of a degree the current position stored in Variable headMR using each call of step 914. If the location left behind to MR list does not exist any more, step 912 sets Variable headMR to null. As for a clean-up generation step, the increment only of 1 is carried out in the order of a scan of JPEG2000 after completion of step 914 to a loop formation 906, as for return and a location P (x y).

[0093] All Cord Brock's locations P within the limits (x y) end the back clean-up

generation step 708 processed by the clean-up generation step 708.

[0094] Thus, while the current position is processed within current SP pass, the location is sent also to the clean-up list generation step 708. At this step 708, the loop formation which passes along all the coordinates in Brock in order of the scan of JPEG2000 exists. The generated coordinate is compared with the next location in current SP coding pass. If the above-mentioned coordinate and a location are equal, the location is beforehand processed within current pass, the processing will be continued, the location of a degree will be acquired in current SP coding pass, and the processing will generate the following coordinate. Processing stands by if the location of a degree obtained from current SP coding pass does not exist until a location is generated, or until SP coding is completed. When a current coordinate is not equal to Variable headSP, a current coordinate exists in the location before the coordinate stored in Variable headSP, therefore this coordinate is the candidate of a clean-up list. Subsequently, this coordinate is compared with Variable headMR. Since the coordinate is already in MR list when the above-mentioned coordinate and a location are not equal, processing acquires the next location in MR list, stores it in Variable headMR, and generates the following coordinate. When this generated coordinate is not equal

to Variable headMR, this coordinate is inserted into a clean-up list. If all locations are encoded, Variable headSP can once be set to null. Similarly, Variable headMR can be set to null when all MR locations of MR list have once been processed. Thus, the coordinate after the last of SP list and MR list can be added to a clean-up list.

[0095] In the example of modification of this configuration, when processing detects that all of four locations of a certain train in a strip are in a clean-up list, processing can compress the four locations into one synthetic coordinate in a clean-up list. This synthetic coordinate can be used in the clean-up phase of next coding/decryption, when it can encode.

[0096] Returning to drawing 7 A, SP coding step 314 returns new SP list and updated original SP list during the clean-up coding pass 318 among the termination step 714. On the other hand, the entropy-code-modulation method 300 progresses to the magnitude refinement MENTO coding pass 316.

[0097] Next, if an eye is changed to drawing 10 , the flow chart of the magnitude refinement MENTO coding pass 316 is further shown in the detail. The magnitude refinement MENTO coding pass 316 begins from step 1002, and progresses to loop formations 1004-1008, and the location in MR list for each

pass of loop formations 1004-1008 is searched in order of the scan of JPEG2000 there. After step 1002, the magnitude refinement MENTO coding pass 316 progresses to steps 1004 and 1006 of a loop formation first, in order to process the 1st location in MR list. After completion of step 1006, the magnitude refinement MENTO coding pass 316 progresses to judgment Brock 1008, and the check of whether the location set as the object of processing which then, was not processed before exists further in MR list is performed. When judgment Brock 1008 returns yes (truth), as for the magnitude refinement MENTO coding pass 316, the next location of MR list is searched [to step 1004] return and there. Among step 1004, the magnitude refinement MENTO coding pass 316 fetches the current position from MR list, and fetches the symbol which corresponds from memory using this current position. The magnitude refinement MENTO coding pass 316 fetches the significant value of the inside near 3x3 of the current position from an effective value table again. The magnitude refinement MENTO coding pass 316 progresses to step 1006 after completion of step 1004. Among step 1006, such effective value is sent out to context generation processing, a context is generated, coding of a corresponding symbol is performed, and algebraic-sign-ization is further performed to the

symbol. These steps 1004 and 1006 can be performed at the rate of one location per 1 cycle. In coding processing, since a required data bit is fetched from bit plane memory, the coordinate of a location is used. In decryption processing, in order to write in a decode bit into the buffer for an output, the coordinate of a location is used. When judgment Brock 1008 returns Nor (false), the magnitude refinement MENTO coding pass 316 is ended (1010), and, subsequently an entropy-code-modulation method progresses to the clean-up coding pass 318.

[0098] Next, if an eye is changed to drawing 11 , the flow chart of the clean-up coding pass 318 is shown. The clean-up coding pass 318 begins from step 1102. the inside of this step 1102, and the last SIG -- the clean-up list generated during the NIFIKANSU propagation coding pass 314 is passed to the current clean-up coding pass 318. Moreover, it is generated by the last clean-up coding pass 308 or 318 inside, and original SP list updated during the last coding pass 314 is passed to the clean-up coding pass 318. Furthermore, new SP list generated during the last coding pass 314 is passed to the clean-up coding pass 318. Among the initiation step 1102, the clean-up coding pass 318 also initializes two lists called the clean-up-significant multiplier list of [for storing the coordinate (a line, train) of a location], and a list near clean-up un-significant, and sets the

entry of those lists to null. Finally, MR list generated in the last clean-up coding pass 308 or 318 is passed to the current clean-up coding pass 318.

[0099] After the initiation step 1102, the clean-up coding pass 318 progresses to loop formations 1104-1114, and the location in a clean-up list is searched during each pass of loop formations 1104-1114 in order of the scan of JPEG2000 there. After step 1102, the clean-up coding pass 318 progresses to judgment Brock 1104, and the check of whether the location which was not searched with the current clean-up coding pass 318 there exists further in a clean-up list is performed. When judgment Brock 1104 returns yes (truth), the clean-up coding pass 318 progresses to step 1106, and the fetch of the next location in the clean-up list which then, was not searched before is carried out. These locations are searched from a clean-up list in order of the scan of JPEG2000. After the fetch of this location is carried out, the clean-up coding pass 318 progresses to step 1108. The clean-up coding pass 318 fetches the bit symbol which corresponds from the bit plane memory of that location among step 1108 using this current position. The clean-up coding pass 318 also fetches effective value from the effective value table near 3x3 of the current position. Furthermore, these significant values are sent out to context generation processing, and the

context for coding of the present symbol is generated. Subsequently, this current symbol is algebraic-sign-ized.

[0100] After completion of step 1108, the clean-up coding pass 318 progresses to judgment Brock 1110, and a check with the current significant bit symbol encoded there is performed. When judgment Brock 1110 returns Nor (false), the clean-up coding pass 318 returns to judgment Brock 1104, in order to process the next location in a clean-up list (when it exists). On the other hand, when judgment Brock 1110 returns yes (truth), clean-up coding pass progresses to step 1112, and the coordinate (a line, train) of the current position is added to a clean-up-significant multiplier list there. The location in a clean-up-significant multiplier list is stored in order of the scan of JPEG2000. The clean-up coding pass 318 progresses to step 1114 after completion of step 1112.

[0101] Among step 1114, the clean-up coding pass 318 determines the un-significant multiplier within the limits near 3x3 of the current position, and adds the location near [these] un-significant to a list near clean-up un-significant. The location of the latter in a list near clean-up un-significant is sorted according to the order of a scan of JPEG2000. A list near clean-up un-significant can have the location of the multiplier which is not in the clean-up

list itself so that clearly [this contractor]. The method of adding these un-significant multipliers to a list near clean-up un-significant is similar with the approach (drawing 13 A, 13B, 14) explained in relation to the addition of an un-significant multiplier on new SP list. Although the same processing as what was explained while the clean-up coding pass 318 referred to drawing 13 A, and 13B and 14 in the case of the former is used, it is accompanied by the following modification. With the clean-up coding pass 318, the coordinate (a line, train) of the significant multiplier encoded now is passed to step 1304 (drawing 13 A) in step 1114, and Variable WinPos is instead added to a list near clean-up un-significant at step 1414. Thus, any overlap of an un-significant multiplier is removed.

[0102] After completion of step 1114, the clean-up coding pass 318 encodes a sign bit according to JPEG2000 (not shown), and, subsequently returns to judgment Brock 1104. When a location does not exist in the clean-up list whose judgment Brock 1104 returned Nor (false) and which was case [the list], namely, searched and processed any more, the clean-up coding pass 318 progresses to step 1116, and ends the clean-up coding pass 318 there.

[0103] while the clean-up coding pass 318 is processing each location in a

clean-up list -- simultaneously, clean-up coding pass -- the following SP coding pass 314 and MR coding pass -- in order to perform processing which follows 316 inside, original new SP list and new MR list are generated (1116).

[0104] Next, if an eye is changed to drawing 12 , the flow chart of step 1116 which generates original new SP list and new MR list is further shown in the detail. Step 1116 begins from 1202 and original SP list updated there, new SP list, original MR list, a clean-up-significant multiplier list, and a list near clean-up un-significant are inputted. Original SP list of [in list memory] is sorted using the significant tag updated among the last coding step 314 (1204), and is changed into two lists. Original SP significant multiplier list is the location of original SP list, and has the location where the tag was attached as a thing with the significant multiplier to which it corresponds in a current bit plane. The latter location is added to a new original MR list. Original SP un-significant multiplier list has the location of original SP list. The tag of this location shows a location with the un-significant multiplier to which it corresponds in a current bit plane. The location of this latter is added to original new SP list. Similarly, new SP list is sorted using the significant tag generated in the last coding step 314 (1206), and is changed into two lists. New SP significant multiplier list is the location of new

SP list, and has the location where the tag was attached as a thing with the significant multiplier to which it corresponds in a current bit plane. The latter location is added to new MR list. It is new SP un-significant multiplier list which has the location of new SP list, and original SP un-significant multiplier list with the tag in which these locations are shown has the un-significant multiplier to which it corresponds in a current bit plane. The latter location is added to new SP list.

[0105] The location of an original SP-un-significant multiplier list, an original SP-significant multiplier list, a new SP-un-significant multiplier list, a new SP-significant multiplier list, original MR list, a clean-up-significant multiplier list, and a list near clean-up-un-significant pass along the inside of two same 'comparison and merge' processings 1208 and 1210 by which all sublists are merged together. The location of an original SP-significant multiplier list, a new SP-significant multiplier list, original MR list, and a clean-up-significant multiplier list are compared and merged by the 1st 'comparison and merge' processing 1208 (on these specifications, it is henceforth called the 1st sublist). The location of an original SP-un-significant multiplier list, a new SP-un-significant multiplier list, and a list near clean-up-un-significant are compared and merged by the 2nd

'comparison and merge' processing 1210 (on these specifications, it is henceforth called the 2nd sublist). The current position of the clean-up list in which current processing is carried out by the loop formations 1104-1114 of the clean-up coding step 318 also receives the 'comparison and merge' processings 1208 and 1210 as an input.

[0106] It investigates which the 'comparison and merge' processing 1210 has in the earliest location in order of the scan of JPEG2000 by accepting one each to one location in the 2nd sublist, and comparing them. Subsequently, the 'comparison and merge' processing 1210 outputs the result to step 1211, and, subsequently accepts the location of a degree from the selected sublist. When two sublists have the same location, both sides are removed and only one is outputted. Although 'comparison and merge' processing 1208 is performed like the 'comparison and merge' processing 1210, it accepts the location from the 1st sublist. Even if there is little this sort TINGU processing per list per 1 clock cycle, things can be carried out, therefore it is [a throughput] desirable for it to generate one location that they are still 1 multiplier / bit per cycle. However, when the current coordinate processed by loop formations 1104-1114 is before the coordinate which 'comparison and merge' processing of these is going to

output, a function suspends this 'comparison and merge' processing. [all] By this processing, lack of the coordinate in MR list is prevented with new original SP.

[0107] The location of a significant un-significant multiplier may exist in the 2nd sublist in fact. These locations are removed from the output of a comparison and the merge processing 1210 among step 1211. Each location outputted by a comparison and the merge processing 1210 is compared with the current significant condition of the corresponding multiplier in that location in the effective value table 1213 among this step 1211. When it is shown by the effective value table 1213 that the multiplier concerned is significant, the location of an un-significant-on assumption multiplier is removed in step 1211. On the other hand, when it is shown by the effective value table 1213 that the multiplier concerned is un-significant, the location outputted by a comparison and the merge processing 1210 is not removed, but forms some original new SP lists.

[0108] Original SP list with new comparison significant step 1211, comparison, and merge processing 1208 and original new MR list are outputted, respectively, and these lists are used with SP coding pass 314 and MR coding pass 316 which follow. Step 1116 which generates a new original SP list and MR list is

ended by 1212 after completion of a list.

[0109] In the example of modification of this configuration, it continues and original new SP / MR list generation processing go into SP coding pass 314 of the next bit plane. When the following SP coding pass 314 starts, it is desirable that the head of original SP list new [by] is ready. Thereby, a stall becomes unnecessary. However, since the stall resulting from clean-up coding pass processing does not arise, list generation processing should progress at the maximum rate.

[0110] It becomes possible to manage the further example of modification of this configuration without the tag of original SP list and new SP list. These lists can be sorted among the list generation step 1116 by reading the effective value of the multiplier corresponding to the location stored in these lists from the effective value table showing whether it is significant in the present bit plane. In the further example of modification, if 1 bit which shows whether a bit is significant is prepared in list memory and that multiplier is processed before, this bit will be written in. This approach is suitable at the reason for reducing access to an effective value table. The above-mentioned bit may be in the memory same as list memory, or may be in separate memory.

[0111] As mentioned above, after completion of the clean-up coding pass 318, this approach returns to judgment Brock 310, in order to process the further bit plane of arbitration. This approach is ended when judgment Brock returns truth (yes) (i.e., when the further bit plane set as the object of processing does not exist) (300).

[0112] Next, if an eye is changed to drawing 15 , the entropy codec for performing the entropy-code-modulation method of illustration to drawing 3 is shown. This entropy codec has the list memory manager 1502, the bit plane splitter 1504, the bit plane memory 1508, the effective value table 1510, the list memory 1506, the context generation logical circuit 1512, and the algebraic-sign-ized section (not shown). The list memory manager 1502 takes the form of an exclusive integrated circuit, and the function is realizing list creation related with the operation of the entropy-code-modulation method of drawing 3 . These lists (for example, new SP list) created by the list memory manager 1502 are stored by the list memory manager 1502 in the list memory 1506 of the form of semiconductor memory. The list memory manager 1502 does not perform the context generation step and coding step of the entropy-code-modulation method itself of drawing 3 , but controls these

actuation performed by the context generation logical circuit 1512 and the codec (not shown). Similarly, the list memory manager 1502 does not perform the bit plane division actuation 304 and actuation 306, but controls these actuation performed by the bit plane splitter 1504.

[0113] A codec 1500 performs the entropy-code-modulation method of drawing 3 as follows. While Cord Brock's wavelet multiplier is inputted from the DWT unit (not shown), the bit plane which constitutes this Cord Brock is written in into each bit plane memory 1508 by the bit plane splitter 1504. The bit plane splitter 1504 determines the bit plane with which each multiplier begins to become significant, and stores the information in coincidence in the effective value table 1510. Moreover, the bit plane splitter 1504 determines whether to be the top bit plane in which which has effective bits in a bit plane, and sends out the top bit plane number to coincidence to the list memory manager 1502. The list memory manager 1502 also fetches the bit plane number of the lowest bit plane set as the object of coding again. the list memory manager 1502 -- from the top bit plane -- starting -- subsequently -- each bit plane -- SIG -- it performs to the lowest bit plane set as the object of coding of the operation of the NIFIKANSU propagation pass 314, the magnitude refinement MENTO pass 316, and the

clean-up pass 308 or 318. With each pass, the list memory manager 1502 creates the location list of all multipliers stored in the list memory 1306 which is all the multipliers that need to perform coding/decode with the pass. SIG -- when the list memory manager 1502 passes along the inside of original SP list in the NIFIKANSU propagation pass 314, new SP list is generated.

[0114] The list memory manager 1502 directs the significant value of the multiplier of the inside near 3x3 of the location which should read from the effective value table 1510 and should be passed to up to the context generation logical circuit 1512, shortly after determining the location of a symbol set as the object of coding. The list memory manager 1502 searches the symbol in the location in a current bit plane from the bit plane memory 1508 again. Subsequently, the context generation logical circuit 1512 calculates a context based on this effective value information, and passes that information to the algebraic-sign-ized section (not shown). Moreover, the symbol in the location in a current bit plane is passed to up to the algebraic-sign-ized section (not shown).

[0115] Next, if an eye is changed to drawing 16 , the flow chart which shows the option 1600 which performs entropy code modulation of the symbol Cord Brock's transform coefficient is shown. This method 1600 of performing entropy code

modulation of a symbol is simpler than the approach of encoding the symbol explained while referring to drawing 3 . However, this approach cannot be used in order to perform an entropy decryption, since this approach 1600 needs to know several bits of a multiplier beforehand. Therefore, although this approach is simple, it can be used only at the time of entropy code modulation.

[0116] This approach 1600 begins from step 1602, and all required parameters are initialized at this step. This approach 1600 generates an effective value table among the initialization phase 1602. This effective value table has the two-dimensional array of the significant condition of the multiplier of Cord Brock set as the object of coding. This approach 1600 processes whole Cord Brock, determines the bit plane in the location where each multiplier begins to become significant, and stores the information in an effective value table. Subsequently, this approach 1600 can determine the significant condition of the multiplier in the present bit plane about the bit plane number stored in the effective value table relevant to the multiplier. For example, when the current bit plane which is related with a certain multiplier and stored in the effective value table is larger than a bit plane number, the multiplier becomes un-significant about the current bit plane. The multiplier becomes significant when a current bit plane is not

larger than a bit plane number. This bit plane number is stored in an effective value table as an array with the coordinate which shows the array corresponding to Cord Brock's multiplier location within the limits (a line, train).

[0117] moreover, the inside of the initialization phase 1602 and this approach -- SIG -- a NIFIKANSU propagation list (SP list), a magnitude refinement MENTO list (MR list), and three lists named on these specifications as a clean-up list (N list) are generated. The location of the bit symbol of Cord Brock encoded within the limits is included in these lists during current pass. for example, this SIG -- SIG current to a NIFIKANSU propagation list -- the location list of all bit symbols of Cord Brock encoded with NIFIKANSU propagation pass within the limits is included. The vocabulary 'the location of a bit symbol' means the array location of Cord Brock of the transform coefficient in which the bit symbol forms a part within the limits on these specifications. Furthermore, the location of each within the limits of these lists is index-ized according to the order of a scan of JPEG2000 (refer to drawing 2). the beginning and SIG -- a NIFIKANSU propagation list (SP list) and a magnitude refinement MENTO list (MR list) are empty. On the other hand, a clean-up list (N list) lists all Cord Brock's locations within the limits, and it is used in order to encode the top bit plane with a non

zero bit.

[0118] Cord Brock's transform coefficient is received after initiation 1602, and it is divided into a bit plane (1604). At the following step 1606, Cord Brock's bit plane is checked and it is judged which bit plane owns a top non zero bit.

[0119] Subsequently, this approach progresses to step 1608 and entropy code modulation of the bit plane which has the most significant bit there is carried out with one single clean-up pass. Entropy code modulation of the bit symbol which specifically belongs to a bit plane with a top non zero bit is carried out respectively. Entropy code modulation of these bit symbols is carried out to drawing 2 in the order of a scan of JPEG2000 like illustration. Reception and this context are suitably determined for the entropy coding step 1608 according to JPEG2000 considering the bit symbol for coding, and the context of the bit symbol as an input.

[0120] After completion of step 1608, this approach progresses to judgment Brock 1610, and the check of whether the bit plane set as the object of processing there exists further is performed. If it exists, this approach will progress to step 1612 and the bit symbol of the next bit plane will be searched there.

[0121] This approach 1600 progresses to step 1613 after completion of step 1612. the inside of step 1613, and this approach 1600 -- first -- SIG -- a NIFIKANSU propagation list (SP list), a magnitude refinement MENTO list (MR list), and a clean-up list (N list) are cleared. then -- step 1613 -- all Cord Brock's locations within the limits -- sorting -- SIG -- these locations are stored in a NIFIKANSU propagation list (SP list), a magnitude refinement MENTO list (MR list), and a clean-up list (N list). The sort step 1613 stores these locations so that each location may be stored only in one of these lists. These stored locations show the coordinate (a line, train) of the multiplier to which it corresponds in Cord Brock.

[0122] Among the sort step 1613, this approach covers all Cord Brock's locations in order of the scan of JPEG2000, and performs an increment.

[0123] The sort step 1613 searches the bit plane number corresponding to the location near the 3x3 which encloses the location under current scan stored in the effective value table during a current location scan. A sort step also searches the bit plane number stored in the effective value table corresponding to the location under current scan again. As mentioned above, the bit plane number stored in the effective value table shows the bit plane which exists in the location

where the multiplier becomes significant. Subsequently, the sort step 1613 determines the significant condition of each multiplier near [including the significant condition of the multiplier of the current position] 3x3 in a current bit plane. Subsequently, the sort step 1613 judges in any of SP, MR, or the N lists the location should be stored according to the following regulations.

[0124] Regulation 1: If the multiplier of the current position P under scan (x y) has significant condition >=N (however, bit plane current [under processing] in N), the current position will progress to MR list.

[0125] Regulation 2 : The multiplier of the current position P under scan (x y) has significant condition smaller (<N) than N. And it is the multiplier which is in front of the current position P (x y) in order of the scan of JPEG2000. If the multiplier of the arbitration of the inside near 3x3 of the current position P (x y) has a multiplier with significant condition >=N (however, bit plane current [under processing] in N), the current position P (x y) will go into SP list.

[0126] Regulation 3 : The multiplier of the current position P under scan (x y) has significant condition (<N) smaller than N. And it is the location of the arbitration of the inside near 3x3 of the current position P (x y) which is behind the current position P (x y) in order of the scan of JPEG2000, and if it has a location with

significant condition >N (however, bit plane current [under processing] in N), the current position P (x y) will go into SP list.

[0127] Regulation 4: The location is stored in N list if the current position P under scan (x y) is not stored in MR list or SP list. That is, the location which is not in SP list and MR list is included in a clean-up (N) list.

[0128] Thus, the sort step 1613 sorts the location of a multiplier and puts it in into SP for bit planes N, MR, and N list. subsequently, these lists -- sending -- SIG -- it can encode in NIFIKANSU propagation pass, magnitude refinement MENTO pass, and clean-up pass.

[0129] after completion of the sort step 1613, and this approach -- step 1614 -- progressing -- SIG of a current bit plane -- processing of the bit symbol related within NIFIKANSU propagation pass is performed. At this step 1614, entropy code modulation of the bit symbol in the current bit plane corresponding to the location in current SP list which is a bit symbol generated at step 1613 is carried out.

[0130] This approach progresses to the magnitude refinement MENTO coding pass 1616 after completion of SP coding pass 1614. At this step 1616, entropy code modulation of all the bit symbols in the current bit plane corresponding to

the location in a current magnitude refinement MENTO list which is the bit symbol generated at step 1613 is carried out.

[0131] This approach progresses to the clean-up coding pass 1618 after completion of MR coding pass 1616. At the clean-up coding step 1618, entropy code modulation of all the bit symbols in the current bit plane corresponding to the location in (it is the bit symbol generated during the last coding pass) and a current clean-up N list is carried out.

[0132] As for this approach, processing of the further bit plane of return and arbitration is performed after completion of the clean-up coding pass 1618 to judgment Brock 1610. When judgment Brock returns truth (yes) (i.e., when the further bit plane set as the object of processing does not exist), it ends (1620) and this approach returns to the call approach. When judgment Brock does not return truth (yes), this approach progresses to step 1612, in order to process the next bit plane. Suitably, as for this approach, it is desirable to continue bit plane processing to the predetermined lowest bit plane.

[0133] Suitably, it is desirable for this approach to generate all three lists, i.e., SP and MR, and N list. In the example of modification of this configuration, this approach can generate at least one list of [of SP, MR, and the N lists]. In this

case, this approach scans whole Cord Brock during the pass of arbitration without a list.

[0134] Next, if an eye is changed to drawing 17 , the entropy encoder which performs the entropy-code-modulation method of illustration to drawing 16 is shown. The entropy encoder 1700 has the bit plane splitter (not shown) for performing the multiplier sorter which sorts the location of the multiplier of the present bit plane according to the regulation mentioned above while referring to drawing 16 , and actuation 1604 and 1606, the bit plane memory (not shown) for storing the bit symbol of each bit plane of Cord Brock, and the effective value table 1710 for storing a significant value. An entropy encoder also has SP list memory 1704 for storing SP list generated by the multiplier sorter 1702 again, MR list memory 1706 for storing MR list generated by the multiplier sorter 1702, and the clean-up list memory 1708 for storing the clean-up list generated by the multiplier sorter 1702. An entropy encoder has further the context generation logical circuit 1712, SP list, MR list, and the algebraic-sign-ized section (not shown) for encoding the bit symbol corresponding to the location stored during each pass of those lists in the clean-up list, as explained referring to drawing 16 . SP list memory 1704 is a double buffer configuration. While the multiplier sorter

1702 generates the location for SP lists of bit planes n and stores those locations in SP list memory 1704, the context generation logical circuit 1712 has read SP list completed for bit planes (n+1). MR list memory 1706 and the clean-up list memory 1708 function also as a double buffer by the same approach as SP list memory 1704 again.

[0135] The above-mentioned desirable approach explained while referring to drawing 3 and drawing 16 has a certain special flows of control. Many other modifications using various flows of control exist in these approaches, without deviating from the pneuma or the range of this invention. For example, the entropy-code-modulation method of drawing 3 has some parallel operation partially. The entropy-code-modulation method of drawing 3 is changed, and such parallel operation is also realizable as if it was the sequential actuation which is clear to this contractor. In the case of the latter (sequential operation), this changed entropy-code-modulation method of drawing 3 is suitable for the implementation as software of a general purpose computer. For example, the pretreatment step 412 and the after-treatment step 418 may function sequentially about some examples of an alteration. The entropy-code-modulation method of drawing 16 may be realized as software of a

general purpose computer.

[0136] Next, if an eye is changed to drawing 18 , the general purpose computer which can perform the above-mentioned approach is shown. Processing of these approaches may be performed as software like the application program performed within the limits of computer system 1800. Especially the step of the above-mentioned approach is performed with the instruction in the software performed by computer. For example, this software may be stored in the computer-readable medium containing the storage explained below. This software is loaded into a computer from a computer-readable medium, and, subsequently is performed by computer. The computer-readable medium with software or a computer program which is saved to a computer-readable medium is a computer program product.

[0137] Computer system 1800 has an output unit containing a computer module 1801, a keyboard 1802 and the input unit of mouse 1803 grade, and a printer 1815 and a display 1814. A computer module 1801 uses the strange recovery (modem) transmission-and-reception equipment 1816 for communicating with the connectable communication network 1820 through the telephone line 1821 or other functional media. A modem 1816 can be used and other network

systems, such as the Internet, a local area network (LAN), and a wide area network (WAN), can also be accessed.

[0138] Generally at least one processor unit 1805, and the I/O interface 1813 for [as the store (memory) 1806 formed from semi-conductor random access memory (RAM) or read only memory (ROM), input/output (I/O) interface including the video interface 1807, a keyboard 1802 and a mouse 1803, and an option] joy sticks (it does not illustrate) and the interface 1808 for modem 1816 are included in a computer module 1801. A store 1809 is offered and, generally a hard disk drive 1810 and the floppy (trademark) disk drive 1811 are contained in this equipment. Magnetic tape transport (it does not illustrate) can also be used. Generally the CD-ROM driving gear 1812 is formed as a non-volatile data source. Generally the components 1805-1813 of a computer module 1801 communicate through the interconnect bus 1804 by the approach well-known to this contractor obtained as a result by the conventional mode of operation of computer system 1800. IBM-PC and a compatible machine, and the same computer system that evolved from Sun Sparstation or there are contained in the example of the computer which can perform the gestalt of operation.

[0139] Generally, the application program of the above-mentioned approach

resides in a hard disk drive 1810 permanently, and is read and controlled by the processor 1805 at the time of program execution. Probably in-between storing of the data of the program by which the fetch was carried out from the network 1820, and arbitration can be performed according to a hard disk drive 1810 using semiconductor memory 1806. the application program read through the driving gear 1812 which is encoded by CD-ROM and the floppy disk and corresponds in some examples, or 1811 -- or the application program which a user can read from a network 1820 through modem equipment 1816 can be supplied to a user. Furthermore, software can also be loaded into computer system 1800 from other computer-readable media including E-mail transmission saved at the radio-transmission channel between equipment different from a magnetic tape, ROM or an integrated circuit, a magneto-optic disk, and a computer module 1801 or an infrared transmission channel, a computer-readable card like a PCMCIA card, a web, etc., the Internet including information, and intranet. An above-mentioned thing is only instantiation of the only related computer-readable medium. It is also possible to actually use other computer-readable media, without deviating from the range and pneuma of this invention.

[0140] The gestalt of operation of this invention it can apply to computer industry, video image industry, and a list at a related field like video and camera industry is clear from the above thing.

[0141] **** [without not passing to the thing explaining the gestalt of some operations of this invention, but deviating from the range and pneuma of this invention, an alteration is performed and an above-mentioned thing can make a change, and / a gestalt / and]. [the gestalt of the above operation] [instantiation]

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is drawing showing the near significant condition of the envelopment multiplier of eight pieces of the multiplier "X [m, n]" used by JPEG2000 of the conventional technique.

[Drawing 2] It is the outline block diagram which illustrates an example of the Cord Brock scanning pattern for Cord Brock used by JPEG2000 of the conventional technique.

[Drawing 3] It is the flow chart which shows the entropy-code-modulation method for following the 1st configuration.

[Drawing 4] It is the flow chart which shows detail processing of the first clean-up coding step 308 used for drawing 3 by the entropy-code-modulation method of illustration.

[Drawing 5] It is the flow chart which shows the pretreatment step 412 of the j-th

train of the i-th strip which is drawn on drawing 4 further to a detail.

[Drawing 6] It is the flow chart which shows the after-treatment step 418 of the train of eye watch (j-2) of the strip of eye watch (i-1) it is drawn on drawing 4 further to a detail.

[Drawing 7 A] **

[Drawing 7 B] SIG of drawing 3 -- it is the flow chart which shows the NIFIKANSU (propagation SP) coding step 374 further to a detail.

[Drawing 8] It is the flow chart which shows step 720 of SP coding step 314 further to a detail.

[Drawing 9] It is the flow chart which shows the clean-up list generation step 708 of SP coding step 314 further to a detail.

[Drawing 10] It is the flow chart which shows the magnitude refinement MENTO coding pass 316 further to a detail.

[Drawing 11] It is the flow chart which shows the clean-up coding pass 318 of drawing 3 further to a detail.

[Drawing 12] It is the flow chart which shows step 1116 for generating original new SP list and new MR list further to a detail.

[Drawing 13 A] **

[Drawing 13 B] It is the flow chart which shows the processing which adds near un-significant to new SP list passed by step 732 of drawing 7 A.

[Drawing 14] It is the flow chart which shows the after-treatment process used with the pretreatment process of drawing 13 A and drawing 13 B.

[Drawing 15] It is drawing showing the schematic diagram of the entropy codec for realizing an entropy-code-modulation method like illustration to drawing 3 .

[Drawing 16] It is the flow chart which shows the entropy-code-modulation method for following the 2nd configuration.

[Drawing 17] It is the schematic diagram of the entropy encoder for realizing the entropy-code-modulation method shown in drawing 16 .

[Drawing 18] It is the schematic diagram of the general purpose computer which performs the approach of drawing 3 and drawing 16 .